Integrating Usability Evaluations into the Software Development Process

Lizano, Fulvio

Publication date:
2014

Document Version
Peer-review version

Integrating Usability Evaluations into the Software Development Process: *Concepts for, and Experiences from, Remote Usability Testing*

By Fulvio Lizano Madriz

Department of Computer Science
Aalborg University

Supervised by Professor Jan Stage

March 24th 2014

List of published papers:

1. Lizano, F., Sandoval, M. M., Bruun, A., & Stage, J. Usability Evaluation in a Digitally Emerging Country: A Survey Study. In Human-Computer Interaction–INTERACT 2013 (pp. 298-305). Springer Berlin Heidelberg (2013)
2. Lizano, F., Sandoval, M. M., Bruun, A., & Stage, J. Is Usability Evaluation Important: The Perspective of Novice Software Developers. In British Computer Society Human Computer Interaction Conference (2013).
3. Lizano, F. & Stage, J. Improvement of novice software developers' understanding about usability: the role of empathy toward users as a case of emotional contagion. Accepted for publication in Proceedings of the 16th International Conference on Human-Computer Interaction (HCII) (2014)
4. Lizano, F., Sandoval, M. M., & Stage, J. Integrating Usability Evaluations into Scrum: A Case Study Based on Remote Synchronous User Testing. Accepted for publication in Proceedings of the 16th International Conference on Human-Computer Interaction (HCII) (2014)
5. Lizano, F. & Stage, J. I see you: Increasing Empathy toward Users' Needs. In Proceedings of the 37th Information Systems Research Seminar in Scandinavia (IRIS 2014).
6. Lizano, F. & Stage, J. Usability Evaluations for Everybody, Everywhere: A field study on Remote Synchronous Testing in Realistic Development Contexts. Accepted for publication in Proceedings of the 8th International Conference on Digital Society (ICDS) (2014)

English Abstract

This thesis addresses the integration of usability evaluations into the software development process. The integration here is contextualized in terms of how to include usability evaluation as an activity in the software development lifecycle.

Even though usability evaluations are considered as relevant and strategic human–computer interaction (HCI) activities in the software development process, there are obstacles that limit the complete, effective and efficient integration of this kind of testing into the software development process. Two main obstacles are the cost of usability evaluations and the software developers' resistance to accepting users' opinions regarding the lack of usability in their software systems.

The 'cost obstacle' refers to the constraint of conducting usability evaluations in the software process due to the significant amount of resources required by this type of testing. Some strategies recommend the use of alternative usability evaluation methods, or an improvement of the usability evaluation process, in order to overcome this obstacle. The 'resistance obstacle' refers to the behavior of software developers who do not accept users' opinions regarding their software. In order to overcome the resistance obstacle, some studies suggested involving software developers, alongside users, in the conduction or observation of usability evaluations.

In this thesis, I am proposing a comprehensive approach for both obstacles by using a synchronous remote usability testing method called remote synchronous testing. By using this method, the evaluators and users are separated while the usability evaluations are conducted. Considering this, my overall research question was: How can remote usability testing contribute to resolving the cost and resistance obstacles by providing an effective and efficient integration of usability evaluations into a software development process? In response to the research question, I conducted two surveys and a series of empirical studies. Six published paper contributions were produced during the PhD project.

My PhD research concluded that the remote synchronous testing method can contribute to resolving both cost and resistance obstacles. In the case of the cost obstacle, the developers can use the method to conduct complete usability evaluations in less time. In terms of usability problems detected, the method has similar results. The developers can integrate usability evaluations into modern software development processes, as the agile methods, by using an iterative scheme of small tests. The efficiency of the method relies on the fact that all the activities in the test process require much less time than at the lab, while still obtaining similar results. In the case of the resistance obstacle, the remote synchronous testing method can provide a practical environment where developers can interact with users in a usability evaluation context. By doing this, the developers improve their understanding of usability and this enables an unconscious contagion process of users' emotions, something which will increase the developers' empathy towards users. The efficiency of the method is grounded in the fact that it allows a remote interaction with users, which facilitates the emotional contagion process.

Keywords: Usability evaluation, software development process, integration of usability evaluation into the software development process, remote usability testing, remote synchronous testing, cost obstacle, actual cost obstacle, resistance obstacle.

Dansk Resumé

Denne afhandling fokusere på integration af brugervenlighedsevalueringer i udviklingsprocessen af software. Integrationen er kontektualiseret I form af, hvordan man inkludere brugervenlighedsevalueringer, som en aktivitet i software udvkilingens livscyklus.

Selvom brugervenlighedsevalueringer er anset, som relevant og strategisk bruger-computer interaktion (HCI), aktivitet i udviklingsprocessen af software, er der forhindringer der begrænser en komplet og effektiv integration af denne type testmetoder. De 2 mest signifikante forhindringer er omkostningerne til brugervenlighedsevalueringer og software udviklerens modstand i at acceptere brugernes meninger verørende manglende brugervenlighed I deres software systemer.

'Omkostnings-forhindringen' refererer til begrænsningen af, udførelsen af brugervenlighedsevalueringer i softwareudviklingen, som følge af den signifikante mængde af ressourcer, som denne type test fordre. Nogle strategier anbefaler brugen af alternative metoder for brugervenlighedsevalueringer, eller en forbedring af processen af brugervenlighedsevalueringen, for at overvinde denne forhindring. 'Modstands-forhindringen' refererer til software udviklernes tilgang, i ikke accepterer brugernes meninger vedrørende deres software. For at overvinde denne modstands-forhindring, har nogle studier foreslået at involvere software udviklerne, sammen med brugerne, i udførelsen og observationerne af brugervenlighedsevalueringer.

I denne afhandling, foreslår jeg en omfattende tilgang til begge forhindringer, ved brug af 'synkron fjern brugervenlighedstest', kaldet 'remote synchronous testing'. Ved brug af denne metode, vil evaluatorer og brugere være adskilt under brugervenlighedsevalueringerne. Ved at tage hensyn til dette, var mit primære forsknings spørgsmål: Hvordan kan fjern brugervenlighedstest, effektivt bidrage til at løse de omkostnings- og modstandsforhindringer, der findes i en softwareudviklingsproces? Som svar på dette spørgsmål, udarbejdede jeg 2 undersøgelser og en række emperiske studier. 6 artikler der omhandler emnet blev produceret gennem PhD projektet.

Min PhD forskning konkluderede at fjern-synkron test metoden, kan bidrage til at løse både omkostnings- og modstandsforhindringerne. Med henblik på omkostningsforhindringerne, kan udviklerne bruge metoden ved at udføre komplette brugervenligheds evalueringer på kortere tid. Med henblik på de fundne brugervenlighedsproblemerne, giver metoden lignende resultater. Udviklerne kan integrere brugervenligheds evalueringer i moderne software udviklingsprocesser, som smidige metoder, ved at anvende en iterativ test. Effektiviteten af denne metode, er baseret på at alle aktiviteterne I en test process, anvender meget mindre tid end i et laboratorieforsøg, alt imens der stadig opnås lignende resultater. I forhold til modstandsforhindringerne, kan fjernsynkron test metode, anspore et praktisk testmiljø, hvor udviklerne har mulighed for at interagere med brugerne i en brugervenligheds evaluerings kontekst. Ved at gøre brug af dette, kan udviklerne forbedre deres forståelse af brugervenlighed og dette kan medføre en ubevidst indvirkning af brugernes følelser, hvilket vil øge udviklerens empati overfor brugerne. Effektiviteten af denne metode er baseret på det fakta, som tillader fjern interaktion med brugere, som derigennem facillitere den emotionelle og ubevidste process.

Nøgleord: Usability, Evaluering af software, Brugervenlighedsevaluering, udvikling af software processer, integration af brugervenlighedsevaluering i software udviklingsprocessen, fjern brugervenlighedsevaluering, fjern synkron evaluering, omkostningsforhindringer, faktiske omkostningsforhindringer, modstandsforhindringer.

Resumen en Español

Esta tesis aborda la integración de las evaluaciones de usabilidad en el proceso de desarrollo de software. La integración se contextualiza aquí en términos de cómo incluir las evaluaciones de usabilidad, como una actividad más, en el ciclo de vida de desarrollo de software.

Aun teniendo en cuenta que las evaluaciones de usabilidad son considerados actividades de Interacción Hombre-Computador importantes y estratégicas dentro del desarrollo de software, hay obstáculos que limitan una integración completa, eficaz y eficiente de este tipo de pruebas en el proceso de desarrollo de software. Dos de los obstáculos principales son el costo de las evaluaciones de usabilidad y la resistencia de los desarrolladores a aceptar las opiniones de los usuarios.

El " obstáculo costo " se refiere a la restricción que existe para la realización de evaluaciones de usabilidad en el proceso de software debido a los muchos recursos requeridos por este tipo de pruebas. Entre otras, algunas estrategias recomiendan superar este obstáculo mediante el uso de métodos alternativos de evaluación de la usabilidad o por medio de la mejora del proceso de evaluación de la usabilidad. El "obstáculo resistencia" se refiere a la conducta de los desarrolladores de software de no aceptar las opiniones de los usuarios sobre la escasa usabilidad en sus sistemas de software. Con el fin de superar este obstáculo algunos estudios sugieren la participación de los desarrolladores de software en la conducción o la observación de las evaluaciones de usabilidad con usuarios.

En esta tesis, estoy proponiendo un enfoque integral para ambos obstáculos mediante el uso de un método de pruebas de usabilidad remoto sincrónico llamado Pruebas Síncronas Remotas. Mediante el uso de este método, las evaluaciones de usabilidad se llevan a cabo estando los evaluadores y los usuarios separados. Considerando esto, mi pregunta general de investigación fue: ¿Cómo puede las pruebas remotas de usabilidad contribuir a resolver los obstáculos de costo y resistencia, proporcionando una efectiva y eficiente integración de las evaluaciones de usabilidad en un proceso de desarrollo de software? Con el fin de responder esta pregunta de investigación, llevé a cabo dos encuestas y una serie de estudios empíricos. Seis contribuciones, artículos publicados, fueron desarrollados durante el proyecto de investigación de doctorado.

Mi tesis doctoral concluyó que el método Pruebas Síncronas Remotas puede contribuir a resolver tanto obstáculo de costo como el de resistencia. En el caso del obstáculo de costo, los desarrolladores pueden utilizar el método para realizar evaluaciones de usabilidad completos en menos tiempo. En cuanto a los problemas de usabilidad detectados, el método ofrece resultados similares a los obtenidos en el laboratorio de usabilidad. Los desarrolladores pueden integrar las evaluaciones de usabilidad, en modernos procesos de desarrollo de software como los métodos ágiles, mediante el uso de un esquema iterativo de pruebas pequeñas. La eficiencia del método se basa en el hecho de que todas las actividades en el proceso de prueba requieren mucho menos tiempo que en el laboratorio, obteniendo similares resultados. En el caso del obstáculo resistencia, el método de Pruebas Síncronas Remotas puede establecer un entorno práctico donde los desarrolladores pueden interactuar con los usuarios en un contexto evaluación de la usabilidad. Al hacer esto, los desarrolladores mejoran su comprensión respecto a la usabilidad y se habilita un proceso de contagio inconsciente de las emociones de los usuarios, algo que después aumentará la empatía de los desarrolladores hacia los usuarios. La eficiencia del método se basa en el hecho de que se permite una interacción remota con los usuarios para permitir el proceso de contagio emocional.

Palabras clave: Evaluación de usabilidad, proceso de desarrollo de software, integración de evaluación de usabilidad en el proceso de desarrollo de software, pruebas de usabilidad remotas, pruebas síncronas remota, obstáculo de costo, obstáculo de costo actual, obstáculo de resistencia.

ACKNOWLEDGEMENTS

This PhD project represents an old dream come true. I would like to give my thanks to a number of people and institutions.

Firstly, I want to say thanks to my supervisor, Professor Jan Stage, who gave me permanent and effective guidance throughout the process. I will always appreciate his patience and thoughtfulness. Secondly, many of my colleagues at The Information System Group also gave their unconditional support. Special thanks to Anders Bruun who always had time to respond to my numerous questions. Finally, my colleague at the Informatics School, National University, Costa Rica, Maria Marta Sandoval helped me in the conduction of many of my experiments. Without her help, this research would not have been possible. Maria Marta, Anders and Jan were co-authors of my papers, for which I would also like to express my gratitude.

I also want to say thanks to the National University (Costa Rica), the Ministry of Science and Technology – MICIT (Costa Rica), the National Council for Scientific and Technological Research – CONICIT (Costa Rica), and Aalborg University for the financial support of this PhD research. Special thanks to Professor Lone Dirckinck-Holmfeld, Dean of the Faculty of Humanities, and Professor Kristian G. Olesen, Head of the Department of Computer Science. Both Lone and Kristian guided me in the initial application process. Lastly, for institutional acknowledgements, I want to recognize the support and help of the heads of the Informatics School, National University – Maria Adilia Garcia and Francisco Mata – as well as the other university authorities who made my doctoral studies possible.

Finally, my family and close friends deserve special recognition in this acknowledgement. Special thanks to my daughter, Pamela, who has been the inspiration of this dream. I will never forget all your real support over the years. Thanks also to you, Marcela. Your love has permanently encouraged me at a distance and when you've been here with me. Special thanks to my friend Heine Grünewald who was the first gentle Dane who received me on my first day at Aalborg. Since that moment, Heine and his wife Heilyn Camacho became my guardian angels here in Denmark. Heilyn, former colleague of mine at Informatics School, National University, and now Assistant Professor at Aalborg University, also guided me in many practical and academic aspects both in the initial application process as well as during my studies. Finally, I've been waiting for this moment for decades. From the bottom of my heart, I want to thank my teachers at primary school – 'Niña' Alicia Madriz, 'Niña' Marta Salas and 'Niña' Flory González – you did a very good job!

This thesis is dedicated to the memory of my beloved father, Omar Lizano. You left me during my PhD studies, but your example of perseverance and tenacity is present in each letter of this thesis.

Contents

1. Introduction

In this chapter I describe the motivation of my thesis, the research questions, the research methods used, and the structure of this document. This chapter introduces some key concepts that will be defined and discussed in Chapter 2.

1.1 Integration of usability evaluation into the software development process

Human–computer interaction (HCI) activities, and particularly usability evaluations, are relevant and strategic activities for software development (Abran et al. 2004). The aim of usability evaluations is to assess the usability of a product in order to identify usability problems and/or collect usability measures (Usability Professionals Association 2012).

The importance of usability evaluations is an undeniable fact. For the users, usability evaluations are important because they result in improved software usability and allow users to become more effective and efficient. The efficiency and productivity of the development organizations can be increased by using the valuable feedback provided by usability evaluations. In addition, software developers can obtain important feedback about the usability of their software. They can improve the software and, at the same time, increase their technical skills. Finally, in the case of the software project, usability evaluations are a key element in software quality assessment because they provide useful feedback that enriches the software design.

The importance of the usability feedback illustrates how the integration of usability evaluations into the software development process is a critical factor for success.

Integration of usability evaluations into the software development process is a concept that can be seen from two perspectives. First, integration can be understood in terms of including usability evaluation as an activity in the software development lifecycle (Ferré et al. 2005). Second, integration can also refer to the use of the feedback obtained during the test (Göransson et al. 2003). This thesis will focus on the first perspective; I am interested in finding a strategy for including usability evaluations, as a regular activity, in the software development process.

1.2 Two Obstacles to integration of usability evaluation

Regardless of the benefits offered by usability evaluations, there are obstacles that limit the complete, effective and efficient integration of this kind of testing into the software development process. Two main obstacles are the cost of usability evaluations and the software developers' resistance to accepting users' opinions regarding the lack of usability in their software systems (Bak et al. 2008).

The 'cost obstacle' refers to the constraint of conducting usability evaluations in the software process due to the significant amount of resources required by this type of testing.

The cost obstacle can take two forms. Firstly, within the development organization, the cost obstacle is normally presented in terms of a "perceived cost obstacle" (Ardito et al. 2011; Bak et al.

2008). Secondly, in the case of the software development project, the obstacle usually takes the form of an "actual cost obstacle" (Nielsen 1994). This obstacle can be measured by defining and collecting information about diverse usability metrics. One of the most recognized metrics is the time consumption in the usability evaluations. (Andreasen et al. 2007; Bruun et al. 2009). There are several strategies aimed at overcoming the cost obstacle in usability evaluations. One of these strategies recommends the use of an alternative usability evaluation method as the inspection method – specifically, heuristic inspection (Nielsen, Molich 1990). This method provides a quick and dirty evaluation of usability. Other strategies address how to improve the usability evaluation process. For example, Kjeldskov et al. (2004) suggest analysis of the usability evaluations by using an alternative analysis method called instant data analysis (IDA).

The 'resistance obstacle' refers to the behavior of software developers who do not accept users' opinions regarding their software. This obstacle reduces the appraisal that software developers have regarding usability evaluations (Ardito et al. 2011; Bak et al. 2008). This obstacle is caused by the coexistence of other obstacles or situations such as the software developers' mindset, the lack of understanding regarding usability (Bak et al. 2008; Rosenbaum et al. 2000; Seffah, Metzker 2004), the developers' low interest in users' needs (Patton 2002) and, finally, the developers' lack of empathy towards users' needs (Grudin 1991). This obstacle can be measured in an indirect way by measuring other obstacles. For example, measuring the lack of understanding regarding usability is possible in order to have an idea of the resistance obstacle. In order to overcome the resistance obstacle, some studies suggest involving software developers in the conduction or observation of usability evaluations alongside users (Hoegh et al. 2006; Gilmore, Velázquez 2000). This strategy improves the developers' understanding of usability and also increases the empathy towards users' needs.

These obstacles seem to have no direct relationship. However, in this thesis, I am proposing a comprehensive approach for both obstacles. Considering this, I will further elaborate these obstacles in Chapter 2.

1.3 Remote usability testing (RUT).

Remote usability testing (RUT) is a method for overcoming these two obstacles. RUT allows software developers to conduct usability evaluations with users in a practical and economical way.

Hartson et al. (1996) defined RUT as a usability evaluation technique in which the evaluator remains separated in space and/or time from the users while performing observation and analysis of the process. RUT can be synchronous or asynchronous. The synchronous format allows the evaluators to receive and conduct the evaluation in real time with users who are located elsewhere. In contrast, in the asynchronous format, the evaluators do not access the data nor conduct the evaluation in real time (Dray, Siegel 2004). RUT allows usability testing without the constraint of geographical limitations, and therefore requires fewer resources. The practicality of logistic considerations, the resource-saving advantage and, finally, the virtual interaction of developers with users, all make RUT a promising alternative for helping to reduce both the cost and resistance obstacles in an economical way.

1.4 Research questions

In this thesis, I will explore an approach for integration of usability evaluations into the software development process which uses RUT to overcome the cost and resistance obstacles presented in the integration. I defined the following overall research question:

RQ: *How can remote usability testing contribute to resolving the cost and resistance obstacles by providing an effective and efficient integration of usability evaluations into a software development process?*

This overall research question has been divided into two further research questions.

The first research question is:

RQ1: *How can software developers use remote usability testing to conduct usability evaluations in an economical way?*

This refers to the overall research question in two aspects. First, it is oriented to pursue a method for resolving the cost obstacle. Second, the method must be efficient in order to help the integration of usability evaluations into a software development process.

The second research question is:

RQ2: *How can remote usability testing reduce the resistance from software developers and make them accept users' opinions about the usability of a software system?*

There are two aspects that connect this question with the overall research question. First, it is oriented to pursue a method for resolving the resistance obstacle. Second, the method must be effective in order to help the integration of usability evaluations into a software development process.

1.5 Research approach.

In order to respond to these questions, I conducted two surveys and a series of empirical studies. In Chapter 4, I will discuss the particular way in which I have used these research methods.

1.6 Structure of the thesis

The thesis consists of five chapters. Following the introduction, Chapter 2 presents the conceptual background for the thesis and a review of the literature to define key concepts and to explore previous studies related to the research questions. In Chapter 3, I present the research contributions, which take the form of six published papers. Here, the contributions, and how they are related, are presented. In Chapter 4, the research methods used in this thesis are presented. Finally, in Chapter 5, I present the conclusions of the thesis by answering the research questions, presenting the limitations and proposing suggestions for future work. The contributions are included in the appendix.

2. Conceptual Background

In this chapter, I present the conceptual background of the thesis. This chapter has two aims: to define key concepts, and to present and discuss literature related to the research questions considered in this thesis.

2.1 Key concepts

In this section, I present the key concepts related to usability evaluation, usability evaluation methods and remote usability testing.

2.1.1 Usability Evaluation

In one of the most comprehensive compilations of knowledge regarding usability (Usability Professionals Association 2012), usability evaluation is defined as:

> *Assessing the usability of a product with the purpose of identifying usability problems and/or obtaining usability measures. The purpose of evaluation can be to improve the usability of the product as part of design/development (formative evaluation), or to assess the extent to which usability objectives have been achieved (summative evaluation).*

In addition, in the Guide to the Software Engineering Body of Knowledge (Bourque, Fairley 2014), 'Usability and Human Computer Interaction Testing' defines the activity in a more detailed way:

> *… [to] evaluate how easy it is for end users to learn and to use the software. In general, it may involve testing the software functions that support user tasks, documentation that aids users, and the ability of the system to recover from user errors.*

The main formative aim of usability evaluations is to provide feedback to the software developers concerning usability problems in order to improve the software (Rubin, Chisnell 2008).

For organizations, usability evaluations are strategic in terms of profitability. If software is easy to use, it is possible to establish a positive relationship between an organization and its customers. For example, usable software can help in the marketing process by providing useful information to the customers and demonstrating that the organization takes care of their needs (Rubin, Chisnell 2008).

From a technical viewpoint, usability evaluations are important for development organizations because they provide historical records about usability for benchmarking purposes. In addition, usability evaluations help to improve the software in order to minimize the cost and time of support for users. Finally, the development organizations can increase their competitiveness and efficiency, and minimize risks associated with the development process (Rubin, Chisnell 2008).

2.1.2 Usability evaluation methods

The majority of usability evaluation methods have been classified into four key types (Usability Professionals Association 2012):

- Usability inspection methods
- Usability testing with users
- Assessment of the use of existing software systems
- Questionnaires and surveys

The first category – usability inspection methods – includes methods used by experienced practitioners in order to assess usability issues. The aim is to provide some useful insights into the software development process. One of the most representative methods in this category is the heuristic evaluation, which is a method where the software interfaces are evaluated based on usability heuristics in order to generate an opinion about the usability of the software (Nielsen, Molich 1990). The process starts with individual reviews by three, four or five expert evaluators of the software. During this process, each evaluator checks if the usability principles, used as a reference for good practices (i.e., heuristics), are included in the software. Next, evaluators compare their results and produce an integral usability report (Holzinger 2005). Other inspection methods are: pluralistic walkthrough, cognitive walkthrough, heuristic walkthrough, metaphors of human thinking (MOT) and persona-based inspection (Usability Professionals Association 2012)

In the second category – usability testing with users – methods are based on the participation of users in order to provide useful feedback related to their experiences while using the software. The classic method of 'usability testing in the laboratory' is normally considered as the clearest example of these methods. Here, the test is conducted by an evaluation staff comprising a test-moderator and observers. The users perform several usability tasks by following the test-moderator's instructions. During the session, each user follows a specific protocol – normally the 'thinking aloud' protocol (Nielsen 2012b) – in order to provide feedback to the evaluation staff regarding their experiences with the software. This feedback is systematically collected and later analyzed in order to produce a list of usability problems (Tullis et al. 2002; Rubin, Chisnell 2008). There are other usability testing methods belonging to this category such as benchmark testing, competitive usability testing, summative usability testing and remote usability testing (Usability Professionals Association 2012).

In the third category – assessment of the use of existing software systems – methods aim to evaluate the usability of existing software and have no relation with a particular software development process. There are different reasons why these evaluations may be required – e.g., auditing the usability of a software system, exploring the reason for changes in the acceptation of a software, etc. In these methods, the aim is to collect feedback directly from experts or users regarding their use of such software. For example, in the 'critical incident report method', the users report the main problems they experienced as a result of the use of the software (Bruun et al. 2009). Once these problems were reported, expert evaluators analyze these reports and produce

usability problem descriptions (Hartson, Castillo 1998). Another usability evaluation method belonging to this category is the 'web analytics method' (Usability Professionals Association 2012).

The last category of methods includes those methods which assess usability by using questionnaires and surveys. An example of these methods is the system usability scale (SUS), which provides a set of statements related to a particular matter. By using this method, the participants must express their degree of agreement or disagreement with each sentence by using a five- or seven-point scale. Results are later quantified in order to analyze a particular status of the usability (Brooke 1996). Other methods belonging to this category are 'rating scales' and the 'satisfaction questionnaire' (Usability Professionals Association 2012).

2.1.3 Remote usability testing

This thesis is specifically focused on the remote usability testing method. Within the diverse definitions of remote usability testing (RUT), it is possible to see several 'keywords' that refer to geographical location, use of technology, separation of evaluators and users, synchronous and asynchronous, etc. Originally, Hartson et al. (1996) defined RUT as a method where:

> *… the evaluator, performing observation and analysis, is separated in space and/or time from the user*

In addition, Menghini (2006) argued that RUT is a technique that:

> *… exploits user home (or office!), transforming it into a usability laboratory where user observation can be done with screen sharing applications.*

Finally, Rubin and Chisnell (2008) defined RUT as:

> *… an option for gathering data about the usability of products where the evaluator is located in one geographic location and the test participant in another.*

RUT has also been defined in terms of those tests which belong to two major categories, synchronous and asynchronous. In synchronous methods, the evaluators can receive and conduct the evaluation in real time with the users who are located in another place. In contrast, the asynchronous format refers to those methods in which the evaluators do not access the data nor conduct the evaluation in real time (Dray, Siegel 2004).

The main uses of RUT are:

- To evaluate the usability of web applications (Menghini 2006).
- To reduce costs of the usability evaluation process (Paternò 2003; Menghini 2006).
- To collect a high volume of data (Dray, Siegel 2004).
- To make usability evaluations by considering an international context (Dray, Siegel 2004).

RUT has various advantages and disadvantages. In Table 1, I present an overview of the main advantages and disadvantages reported in the literature.

Table 1. Advantages and disadvantages of RUT

	Advantages	Disadvantages
Settings and technology	• Reduced interference • Diversity of geographical areas • Easy data collection • Freedom of action • Diversity of technology • Quicker setup	• High planning • Lack of control of the user environment • Technical problems • Security risks • System and connection performance issues • Users' time constraints • Limited visual feedback
Ethnography	• Realistic environment • Better understanding of users' real experience • Use of 'real-life' devices	• Ethnographic legitimacy • Unclear cultural context • Lack of deep understanding of the users' behavior
Actors	• Audience diversity • Easy recruiting of participants	• Recruiting validity • Interpersonal dynamic of the process • Lack in the control of the users' behavior • Resistance to participation
Resources	• Cost savings • Time savings	• Not necessarily inexpensive

In terms of settings and technology, RUT has several advantages. One of the most prevalent is that RUT reduces unnecessary interference with users by the members of the evaluation team (Bartek, Cheatham 2003). In addition, RUT is a technique which allows access to a wide diversity of geographical areas (Dray, Siegel 2004) by collecting data in quite an easy way (e.g., surveys, questionnaires, detailed and scalable usability metrics – as, for example, patterns of users' preferences, etc.) (Albert et al. 2009). Other advantages are: freedom of action, quicker setup (Bolt et al. 2010) and diversity of technology (Fidgeon 2011).

RUT has disadvantages associated with settings and technology. One of the most important disadvantages is that usability evaluations conducted with RUT demand high levels of planning (Albert et al. 2009). In addition, there is a lack of control of the user environment (Bartek, Cheatham 2003). This problem is relevant because the evaluations conducted with RUT require a high level of communication, thus increasing the complexity of the process control. Furthermore, technical problems and security risks are other disadvantages related to the previous disadvantage. The technical problems can increase the problems associated with the lack of control over the user environment. At the same time, the security risks may be increased, especially in those matters related to the information management process (Bolt et al. 2010). Other disadvantages are: system and connection performance issues, limited visual feedback (Bartek, Cheatham 2003) and users' time constraints (Albert et al. 2009).

An important group of advantages is related to ethnographic matters. Usability evaluations conducted with RUT allow realistic environments and unrestricted interaction with participants located in different locations, in real time, and regardless of distance (Bartek, Cheatham 2003). This interaction can provide a better understanding of users' real experiences (Albert et al. 2009). Furthermore, by using RUT, it is possible to use 'real-life' devices (Fidgeon 2011) by providing interesting ethnographic possibilities.

The most evident disadvantage related to ethnographic matters is the legitimacy of RUT (Bolt et al. 2010) – not only because it is a technique which omits physical presence but also due to RUT being a relatively novel technique which needs further development. In addition, Dray and Siegel (2004) argue that some results of the usability evaluations conducted with RUT can be limited due to an unclear cultural context, a lack of appreciation of the 'body language', indirect cues, etc. This limitation leads to other problems in the lack of deep understanding of the users' behavior presented in RUT processes (Albert et al. 2009).

RUT also has advantages related to the tests' actors. By using RUT, it is possible to include a wide diversity of participants as well as members of the evaluation staff (Fidgeon 2011). In addition, RUT methods allow quite a simple recruiting process by using technological tools (Albert et al. 2009).

There are several disadvantages of RUT associated with the tests' actors. Firstly, from an alternative viewpoint of the recruiting process, Bolt et al. (2010) argue that RUT can have problems related to the lack of recruiting validity due to the non-personal treatment or attention given to potential participants in the tests. Secondly, the interpersonal dynamic of the process is another disadvantage due to the lack of face-to-face contact between the actors (Dray, Siegel 2004). Thirdly, the lack of control of the users' behavior, which can be increased precisely for the interpersonal dynamic, is another disadvantage presented in RUT (Bolt et al. 2010). Finally, in RUT, it is possible to find certain resistance to participation in the tests (Bolt et al. 2010).

In terms of economic matters, the advantages and disadvantages are quite obvious. The main advantages of RUT are the cost and time savings (Bartek, Cheatham 2003; Fidgeon 2011; Dray, Siegel 2004; Albert et al. 2009; Bolt et al. 2010). The main disadvantage of RUT is an alternative viewpoint regarding the cost matter. Usability evaluations conducted with RUT are not necessarily inexpensive (Bolt et al. 2010). Even considering that there are some savings of resources in transportation of the evaluation staff, allowances, etc., the truth is that the test process must be conducted considering a similar approach in terms of procedure (i.e., guidance to users, analysis of results, documentation of the process, etc.)

2.2 Integrating usability evaluations into the software development process

In this section, I present the integration concept, the relevance of integration, and the obstacles for integration: the cost obstacle and the resistance obstacle.

2.2.1 The integration concept

Integration of usability evaluations into the software development process has two main aspects. The first aspect refers to how usability evaluations fit into the software process. In this case,

integration means to put the usability evaluation(s), as regular activity, or activities, into a specific place, or places, in the software development lifecycle. An example of this approach is provided by Ferré et al. (2005), who studied the integration of usability evaluations in an iterative software process. They proposed a framework that characterized 35 selected HCI techniques (including usability evaluations) in relation to six important criteria from a software engineering viewpoint. The aim of this framework is to precisely define where the HCI techniques must be considered.

The second aspect concerns the integration of usability evaluations in terms of using their results to improve the software as well other aspects of the software development process. Here, integration not only includes usability evaluation activity in the lifecycle, but moves forward in order to consider the diverse usability techniques as natural components of the development process. The integration here is more oriented towards using the results of the usability evaluations as feedback, which enriches the development process. This approach to integration can be focused on specific phases of the software development process, as was suggested by Göransson et al. (2003), for the design phase. Alternatively, it may consider wider scopes, as was proposed by Hvannberg and Law (2003) in their Classification of Usability Problems (CUP) Scheme, which allows an improvement of the user interface, improves the software development process and increases personal skills (Hvannberg, Law 2003).

In this thesis, I will focus on the first aspect cited; I am interested in finding a strategy that facilitates the use of usability evaluations as a regular activity in the software process.

2.2.2 Relevance of integration

Integration of usability evaluations is an important matter that can be analyzed by considering:

- Users
- Software developers
- Development organizations
- Software development projects

In the case of the user, who requires a high level of usability in the software (Lindgaard, Chattratichart 2007), usability evaluations are important because they assess if the software under evaluation considers users' skills, experiences and expectations (Bourque, Fairley 2014). A high level of usability in a software system enables users to perform their work while saving time and resources, and this is something that allows them to be more effective and efficient.

In the case of software developers, usability evaluations provide them with clear details about the usability problems in a software system. This information becomes valuable feedback (Hoegh et al. 2006), which allows them to produce better results in their work. Furthermore, improved usability in the software increases the developers' confidence levels regarding their technical ability and creates a personal identification with a software product – these are strong motivators for developers (Rasch, Tosi 1992; Hertel et al. 2003).

For the development organizations, usability evaluations are important because they provide benefits such as cost savings (Nielsen 1993), increased sales, productivity, lower training costs and the reduction of technical support for users (Bak et al. 2008). More usable software implies less user support and training, which will increase the development organizations' efficiency and productivity.

Finally, in software development projects, HCI techniques have a high valuation (Jia 2012). In fact, Abran et al. (2004) consider that usability evaluations are relevant and strategic activities within software projects. One of the main reasons for this high valuation is due to the application of usability methods (e.g., usability inspection methods, usability testing with users, etc.), and it is possible to improve the quality of the software by providing useful feedback about usability.

The importance of usability in the above cited cases has motivated the integration efforts of usability evaluations into software projects (Juristo, Ferré 2006; Radle, Young 2001; Granollers et al. 2003). Indeed, the necessity of this integration has been identified in modern methodologies such as agile software development (Sohaib, Khan 2010).

Despite these facts, a full integration of usability evaluations into software projects faces significant challenges. Some usability evaluation methods demand expensive and elaborate procedures, complex logistical considerations, participation of users, long periods of time to analyze tests' results, etc. (Nielsen 1993; Nielsen 1994; Rubin, Chisnell 2008). On the other hand, both software developers and human–computer interaction (HCI) practitioners normally have different perceptions regarding software development (Ferré et al. 2001). This is something that represents profound differences of opinion between these professionals regarding how internal usability testing should be conducted (Jerome, Kazman 2005).

2.2.3 Obstacles for integration

The Oxford Advanced Learner's Dictionary defines an obstacle as "a situation, an event, etc. that makes it difficult for you to do or achieve something" (Wehmeier 2007).

The situations or events that limit the integration of usability evaluations into the software development process can be measured by using a framework based on the total cost of ownership (TCO) approach (Brereton 2005; Airaksinen, Byström 2007). TCO is a comprehensive assessment of information technology over time. This approach advocates consideration of the inclusion of all costs associated with the development process – in this case, the software system.

There are two types of cost in the TCO approach: direct and indirect (Airaksinen, Byström 2007). Direct costs include those costs related to work and capital costs. In terms of usability evaluations, it is possible to identify some examples of this kind of cost in the hourly cost rate of the individuals who have participated in the test (i.e., evaluation staff and users), the cost of renting the usability lab, etc. The indirect costs are more difficult to define due to their unmeasurable nature – e.g., the stakeholders' satisfaction with the usability evaluation testing process, the impact of usability evaluation feedback in the software development process, etc. However, these indirect costs are important. Airaksinen and Byström (2007) argued that as much as 60% of total costs for management and ownership of information technology are indirect costs.

Based on the TCO's categorization of direct and indirect costs, it is possible to consider two main categories of obstacles. The first category includes the 'measurable obstacles' that, similarly to the direct costs in the TCO approach, can be measured directly and in a relatively simple way. The most representative example of this category is the cost of usability evaluations (Ardito et al. 2011; Bak et al. 2008; Nichols, Twidale 2003; Nielsen 1994; Rosenbaum et al. 2000).

The second category is that of 'unmeasurable obstacles'. As with the indirect costs used in the TCO approach, it is necessary to use indirect ways to measure these obstacles. An example of an unmeasurable obstacle is the developers' resistance to accepting the users' opinions (Bak et al. 2008).

In Table 2, I present some of the most representative obstacles to integration.

Table 2. Obstacles in the integration of usability evaluation into the software development process

	Measurable obstacles	Unmeasurable obstacles
User	• Users or customers participation and/or availability	• Resistance to participation
Software developer	• Development and use of personal 'computer-assisted usability engineering tools' • Lack of competence • Lack of understanding	• Software developers' mindset • Resistance to acceptance of the users' opinions • Resistance to UCD and usability
Development organization	• Lack of understanding • Lack of trained engineers in usability/HCI	• Cost (perceived) • Lack of communication of the impact of results • Resistance to user-centered design (UCD) and usability • Lack of respect and support for HCI's practitioners • Lack of 'usability culture'
Software project	• Cost (actual) • Difficult conduction of the testing process • No suitable methods • Limited description of HCI's topics in software engineering (SE)	• Gap between usability and software development process

For the users, participation and/or availability can limit the usability evaluations made in a software system under development (Ardito et al. 2011; Bak et al. 2008). This obstacle can be quantified by calculating the cost of the participation of the originally scheduled users. This cost represents the expense that would be incurred should replacement users be required to replace those originally scheduled.

The main unmeasurable obstacle in the case of the users is the resistance to participation in the usability evaluations (Bolt et al. 2010). In order to have an idea about the impact of this obstacle, it

is possible to evaluate historical records of previous software development processes. The aim is to identify references to the users' participation in such records. The analysis of users' participation should provide an idea of the potential commitment of users and, consequently, to understand what can reasonably be expected in relation to the users and their participation in usability evaluations.

In the case of software developers, there are three measurable obstacles. The first obstacle is the development and use of 'computer-assisted usability engineering tools'. Seffah and Metzker (2004) found that developers occasionally build their own computer-assisted usability engineering tools, which are personal perceptions about usability that may increase the gap between HCI and SE. The second obstacle is the lack of competence (Andreasen et al. 2006) that some developers have in terms of eventually participating effectively in usability evaluations. The third obstacle is the lack of understanding regarding usability (Bak et al. 2008; Rosenbaum et al. 2000; Seffah, Metzker 2004). All of these obstacles can be measured by assessing the knowledge of the information technology staff regarding usability evaluation's main concepts and applications. Simple assessment techniques, as used in academia, can provide a clear idea about the status of these obstacles and infer their impact.

In the case of software developers, there are several unmeasurable obstacles. Having an idea about the impact of these obstacles implies a complex process. The first obstacle is the software developers' mindset (Ardito et al. 2011; Bak et al. 2008). In order to study the impact of this obstacle it is possible to collect opinions of developers, and assess the relative importance that developers give to users, their needs, etc. The second obstacle is the resistance to accepting the users' opinions (Bak et al. 2008), which can be explored by analyzing the measurable obstacle 'lack of understanding'. The third obstacle is the resistance to UCD and usability (Rosenbaum et al. 2000). This obstacle can be dimensioned using a similar strategy followed in the case of the software developers' mindset.

In the case of the development organizations, the first measurable obstacle is the lack of understanding regarding usability evaluations (Bak et al. 2008; Rosenbaum et al. 2000; Seffah, Metzker 2004), which can be measured in a similar way to that used with the measurable obstacles of the software developers (i.e., using academic assessment techniques). The other measurable obstacle is the lack of trained engineers in usability/HCI that limits integration (Rosenbaum et al. 2000). This obstacle results from having untrained staff, and the development organizations are forced, at least initially (Bruun 2013), to hire external experts to conduct usability evaluations. The cost of hiring these external experts can be considered as the measure of the impact of this obstacle.

On the other hand, there are several unmeasurable obstacles such as the perceived cost of the usability evaluations (Ardito et al. 2011; Bak et al. 2008; Rosenbaum et al. 2000), a lack of communication of the impact of usability evaluations (Rosenbaum et al. 2000), a resistance to UCD and usability (Rosenbaum et al. 2000), a lack of respect and support for HCI practitioners (Gulliksen et al. 2004) and, finally, a lack of "usability culture" (BugHuntress 2007; Muzaffar et al. 2011).

Within these obstacles, it is possible to find indirect ways of having an idea about their impact. For example, in the case of the lack of respect and support for HCI practitioners, it is possible to compare the training budget of HCI practitioners against other software engineering practitioners.

In the case of software projects, the majority of the obstacles are measurable. The first obstacle is the cost of the usability evaluations. In contrast to the 'perceived cost obstacle' shown in the case of the development organizations, in the case of the software projects, the cost of the usability evaluations is seen as an 'actual cost' (Ehrlich, Rohn 1994) because it is possible to measure it more precisely. The rest of the obstacles can be measured in a similar way to the lack of competence obstacle presented in the reference to software developers (i.e., using academic assessment techniques). The second obstacle is the difficult conduction of the testing process (Bak et al. 2008). The third obstacle is the absence of a method for conducting the tests (Ardito et al. 2011). Finally, the fourth obstacle is the limited description of HCI's topics in software engineering (SE) (Ferré et al. 2006).

Finally, in software projects it is possible to find an unmeasurable obstacle related to the gap between usability and the software development process (Seffah, Metzker 2004). In order to have an idea about the impact of this obstacle, it is possible to compare the perceptions of developers regarding the main usability activities used in a typical software development process. These perceptions can be later contrasted against accepted benchmarks (e.g., standards, common accepted knowledge, etc.). Differences can expose the magnitude of the gap.

In this thesis, I have focused on two obstacles: the cost of usability evaluations and the software developers' resistance to accepting users' opinions. For the cost obstacle, my interest lies in the actual cost within a software project context. In the remaining two sections of this chapter, I will present the literature related to these obstacles.

2.2.4 The cost obstacle

The obstacle regarding the 'cost of usability evaluation' is related to the constraints of applying usability evaluations due to the high consumption of resources required by this kind of testing. The cost of usability evaluations is a measurable obstacle presented in both cases: development organizations and software projects.

In the case of the development organizations, this obstacle is presented in the form of a 'perceived cost obstacle'. In this case, the perception can be understood as the perspective that the development organizations have regarding the cost of the usability evaluations. This perspective is normally based on the value judgment presented within the development organizations. Some examples of this modality of cost obstacle were reported by Ardito et al. (2011) and Bak et al. (2008) who found that in development organizations there exists the idea that usability testing is expensive, and this limits its application – even though they have not conducted such evaluations. In addition, Nielsen (1994) argues that the perception of the cost of usability engineering techniques is the reason why such techniques are not used extensively in development organizations.

Coincidentally, Bellotti (1988) reports that software developers view usability methods as too time consuming and intimidating in their complexity.

Within software projects, the cost obstacle appears in the form of an 'actual cost obstacle'. Considering the dynamic presented in the software development project, based on a specific product (i.e., the software), the cost obstacle is more tangible and is related to the real cost presented in such a project. Nielsen (1994) offers some examples of actual costs. Ehrlich and Rohn (1994) referred to the actual costs in terms of 'initial costs' and 'sustaining costs'. The initial costs include the settings and the equipment of the laboratories, or similar facilities, that are required for usability evaluations. The sustaining costs correspond to those costs related to the conduction of the usability evaluation process and include staff, recruitment of participants, transportation, allowances, special equipment and software, etc.

As a measurable obstacle, the cost obstacle can be quantified by defining and collecting information about diverse usability metrics. The time consumption in the usability evaluations is one of the most commonly used measures to assess their 'cost' (Andreasen et al. 2007; Bruun et al. 2009; Jeffries et al. 1991; Kantner, Nielsen 1994). Time consumption gives some idea about the consumption of resources in usability evaluations. For example, based on this measure, some studies concluded that classical protocols, such as 'thinking aloud', have high consumption of time (Alshaali 2011; Holzinger 2005). In addition, Kjeldskov et al. (2004) found that analysis of the data collected during the usability evaluations normally demands a high time consumption, especially in the video data analysis process. Finally, Borgholm and Madsen (1999) argue that usability reports could be impractical due the extensive time used in their preparation.

It is possible to identify two main strategies for reducing the cost of usability evaluations:

- Use of alternative usability evaluation methods
- Improvement of the usability evaluation process

Alternative usability evaluation methods aim to reduce costs in classical usability evaluations with users. One example of these methods is the heuristic evaluation (Nielsen, Molich 1990). This strategy is not considered in this PhD project.

The approach of improving the usability evaluation process has been widely discussed. For example, the time consumption issue within analysis activities was addressed by Kjeldskov et al. (2004). They proposed an analysis technique called instant data analysis (IDA) that is used in the analysis process of the results of the sessions with users. The aim of IDA is to conduct usability evaluations in one day, obtaining similar results to traditional video data analysis methods. Alternatively, Borgholm and Madsen (1999) suggest focusing on the report of the results of the usability evaluations. These researchers found that some HCI practitioners prepared two kinds of reports with different formats and contents. The first report was oriented to the developers and provided an executive summary with information useful for their work. The second report, which is more extensive, was delivered several days after the evaluation for documenting purposes. Supplementary meetings, at which the

developers and HCI practitioners discussed the usability findings, and posters, which described the main usability problems, were used to handle this problem.

2.2.5 The resistance obstacle

The resistance obstacle refers to the behavior of software developers who do not accept the users' opinions about their software, and this is something that reduces the valuation software developers have regarding usability evaluations. This obstacle is caused by the coexistence of other obstacles or situations:

- The software developers' mindset
- The lack of understanding regarding usability
- The developers' low interest in users' needs
- The developers' lack of empathy toward users

The software developers' mindset is characterized by the developers' interest in the functionality and efficiency of the code. For some developers, it is difficult to think like users (Ardito et al. 2011; Bak et al. 2008). Personal ability and a sense of pride regarding their software are two of the main motivators for developers (Rasch, Tosi 1992; Hertel et al. 2003). The developers' mindset collides frontally with the results of usability evaluations because these results represent an entirely different perspective to those expected by developers regarding their work. The developers' mindset is an unmeasurable obstacle (see Table 2).

The lack of understanding regarding usability exhibited by some developers (Bak et al. 2008; Rosenbaum et al. 2000; Seffah, Metzker 2004) contributes to the resistance obstacle. This lack of understanding prevents developers clearly understanding the relevance and goals of usability evaluations in software projects. The lack of understanding is a measurable obstacle (see Table 2) and can help in approximating the status of the resistance obstacle. The lack of understanding can be measured by assessing the status of the developers' knowledge of the HCI domain, especially in those aspects related to the users' needs and their work.

The developers' low interest in users' needs is one of the results of the lack of understanding. This low interest was reported by Patton (2002), when he argued about extreme programming's inability to truly connect with customers' needs. An absence of interest in users forms a misunderstanding of their needs, and this is the first sign of a failed software project (Reel 1999) and also reveals a low level of empathy towards users and their needs.

Finally, the developers' lack of empathy towards users is the last element presented in the resistance obstacle. A lack of empathy or sympathy from software developers for inexpert or non-technical users was cited by Grudin (1991) as an obstacle to users' involvement in software projects In fact, Newell et al. (2006) emphasize the importance by the generation of empathy with users during the interface design.. Empathy towards users' needs is the last element presented in the resistance obstacle, and it is also the key concept in those strategies proposed to overcome this obstacle.

An improvement of the empathy towards users was found by Hoegh et al. (2006) when developers reviewed usability reports and when they observed usability evaluations. Coincidentally, Gilmore and Velázquez (2000) argue that developers' participation in user experiences is important in order to improve the software developer's empathy for users. Greater levels of involvement of software developers in usability evaluations (e.g., by conducting usability evaluations), are not only feasible (Skov, Stage 2012), but they also help to overcome the software developers' mindset (Bruun, Stage 2012) and reduce the lack of understanding (Hoegh et al. 2006).

Empathy is defined in terms of the capacity of an observer to observe, identify, and understand another person's feelings, thus producing in this observer, at least partially, a sense of sharing the same feelings of the other person (Decety, Jackson 2006; Singer, Lamm 2009). According to Decety and Jackson (2006), empathy is characterized by three components. The first is an eventual affective response, which the observers have in order to share the other feelings. The second element is a cognitive capacity to adopt the perspective of the observed. Finally, the observer can regulate his/her own emotions.

Empathy can be explained as the result of an emotional contagion (EC) process experienced by the observer when he/she sees another person (Singer, Lamm 2009). This process takes place before emotional empathy and, later, cognitive empathy (De Vignemont 2004).

The EC theory has been investigated in group psychology. Rapson et al. (1993) define EC as:

> *The tendency to automatically mimic and synchronize expressions, vocalizations, postures, and movements with those of another person's and, consequently, to converge emotionally*

Similarly, Schoenewolf (1990) defined EC as:

> *…a process in which a person or group influences the emotions, or behavior of another person or group through the conscious or unconscious induction of emotion states and behavioral attitudes*

According to Barsade (2002), EC is inherent to social influences, which occur both consciously and subconsciously. Another interesting aspect of EC is that it initially involves the perception of emotions using, firstly, nonverbal signals – such as facial expressions, body language and tone – rather than words. Emotions can be transmitted to others in order to influence not only their emotions but also the perception of other aspects. For example, Pugh (2001) shows that when employees display a positive emotion towards customers, these customers perceive more quality in the service obtained.

Considering that the EC process is normally produced in an automatic or unconscious way, there are relevant and challenging considerations which must be taken on board in order to study the origin of empathy of software developers towards users when these developers are involved in usability evaluations. Empathy, and its relation with usability, has been studied mainly in user-centered design (Mattelmäki, Battarbee 2002; Koskinen, Battarbee 2003; Sotamaa et al. 2005; Fulton Suri 2003; Dandavate et al. 1996).

The key concepts of usability evaluations and the research questions have been presented. The next chapter will present the contributions included in this thesis.

3. Contributions

This chapter presents the contributions included in this thesis. These contributions are aimed at responding to my research questions and are as follows:

1. Lizano, F., Sandoval, M. M., Bruun, A., & Stage, J. Usability Evaluation in a Digitally Emerging Country: A Survey Study. In Human-Computer Interaction–INTERACT 2013 (pp. 298-305). Springer Berlin Heidelberg (2013)
2. Lizano, F., Sandoval, M. M., Bruun, A., & Stage, J. Is Usability Evaluation Important: The Perspective of Novice Software Developers. In the British Computer Society Human Computer Interaction Conference (2013).
3. Lizano, F. & Stage, J. Improvement of novice software developers' understanding about usability: the role of empathy toward users as a case of emotional contagion. Accepted for publication in Proceedings of the 16th International Conference on Human-Computer Interaction (HCII) (2014)
4. Lizano, F., Sandoval, M. M., & Stage, J. Integrating Usability Evaluations into Scrum: A Case Study Based on Remote Synchronous User Testing. Accepted for publication in Proceedings of the 16th International Conference on Human-Computer Interaction (HCII) (2014)
5. Lizano, F. & Stage, J. I see you: Increasing Empathy toward Users' Needs. In Proceedings of the 37th Information Systems Research Seminar in Scandinavia (IRIS 2014).
6. Lizano, F. & Stage, J. Usability Evaluations for Everybody, Everywhere: A field study on Remote Synchronous Testing in Realistic Development Contexts. Accepted for publication in Proceedings of the 8th International Conference on Digital Society (ICDS) (2014)

Figure 1 shows the relation between the contributions, the research questions and the research setting.

In my research, I studied how to overcome two important obstacles for integrating usability evaluations into the software development process: the cost and resistance obstacles. This corresponds to the two research questions presented in Section 1.4 and the columns in Figure 1.

There were several settings involved in the research process. The surveys (contributions 1 and 2) were conducted considering diverse development organizations and advanced students with practical experience similar to novice software developers. The experiment in the lab (contribution 3) was conducted in a usability lab with participation of students in software engineering and computer science. The case study (contribution 4) was developed in a software development project made in an academic unit of a university. The field experiments (contributions 5 and 6) were conducted on real software projects in diverse organizations (e.g., schools, colleges, biological research organizations, municipal police stations, etc.).

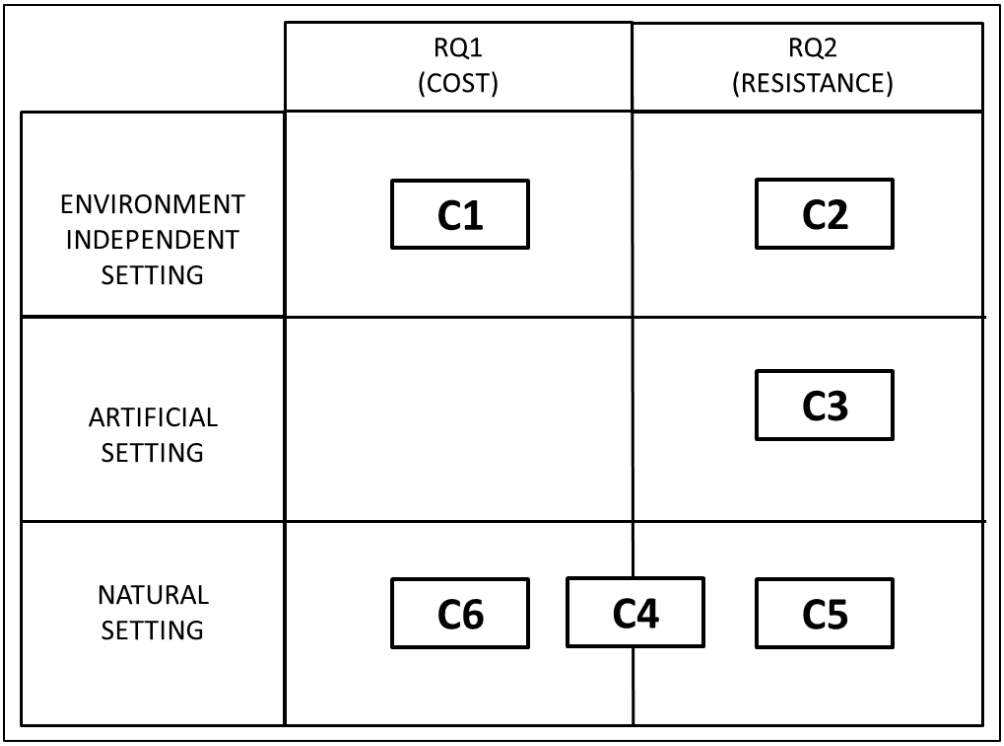

Figure 1. Relation between contributions, research questions and research setting

In the following subsections, I will elaborate on each contribution.

3.1 Contribution 1

Lizano, F., Sandoval, M. M., Bruun, A., & Stage, J. Usability Evaluation in a Digitally Emerging Country: A Survey Study. In Human-Computer Interaction–INTERACT 2013 (pp. 298-305). Springer Berlin Heidelberg (2013)

This contribution provides the results of a survey that explored the application of usability evaluation in software development organizations. The study explored the understanding of the usability evaluation concept, and the advantages and obstacles of the applications of usability evaluations according the cost perspective of the organizations. The contribution presents the results of the study and compares these results with other studies (Ardito et al. 2011; Bak et al. 2008).

The study was carried out in Costa Rica and involved organizations which potentially conducted usability evaluations and met the specific criteria – i.e., located in a specific geographical area, developed software or hardware with graphical user interfaces, developed software for customers or for internal use, and finally, employed more than a single person. The organizations were contacted through the list of organizations affiliated to a chamber of information and communication technology organizations. The questionnaire used in the study contained the following main parts: demographic and general information, products/services provided by the development organization, methodology used to develop software applications, understanding of

the usability evaluation concept, and obstacles and advantages of the application of usability evaluation. A combination of open and closed questions was used to collect the information. Different analytical approaches were used in the study. A quantitative analysis was used on the closed questions, and grounded theory (Strauss, Corbin 1998) was used for the analysis of the open questions.

The findings show that usability evaluations are conducted extensively by the development organizations that participated in the study. About 80% of them reported conducting this kind of test. In addition, in these development organizations, there exists a fairly clear understanding of the meaning of usability evaluation and advantages and obstacles were demonstrated that were similar to those found in other studies. By using an open question, the study revealed that 11% of the participants think that the cost of the usability evaluation process is an obstacle. Furthermore, through a closed question, 18% of the participants considered that resource demands form an important obstacle for applying usability evaluations. In the same closed question, other obstacles were shown to be the recruitment of test participants (20% of the participants) and the conduction of the test or absence of method (13%)

The contribution helps to answer the first research question related to the cost obstacle, because it confirms that, in the context where the main experiments of the PhD project were conducted, the cost obstacle is also presented, thus confirming previous studies.

3.2 Contribution 2

Lizano, F., Sandoval, M. M., Bruun, A., & Stage, J. Is Usability Evaluation Important: The Perspective of Novice Software Developers. In the British Computer Society Human Computer Interaction Conference (2013).

This contribution provides the results of a survey that explored the perspective of novice software developers regarding usability evaluation. The study explored the understanding of the usability evaluation concept, and the advantages and obstacles of the applications of usability evaluations as they were seen by novice software developers. In addition, the study also explored the extent to which novice developers considered usability important, in a general sense. To complement this perspective, the contribution presents the results of the study and compares these results with other studies (Ardito et al. 2011; Bak et al. 2008; Rosenbaum et al. 2000; Seffah, Metzker 2004).

The survey used an online questionnaire with the participation of advanced students of a system engineering undergraduate course. The students were contacted through the official course list of students and projects. The questionnaire was divided into sections such as demographic and general information, importance of usability, understanding of the usability concept, and the obstacles and advantages of usability evaluation. Data on obstacles and advantages of usability evaluations were collected by using a combination of open and closed questions. Different approaches to analyses of the collected data were used; quantitative analysis was used for the closed questions, and the grounded theory approach (Strauss, Corbin 1998) was used for the open questions.

There are three main findings in this contribution. Firstly, the study confirms that for novice software developers, the technical software development activities are more important than the usability activities. The overall average perceived importance was 61% for the software development activities, and this contrasted with 39% for usability activities. Secondly, despite this fact, there was a clear understanding of the usability evaluation concept. Thirdly, the novice software developers believe that they do not obstruct the application of usability evaluations. According to the novice developers, the main obstacle for applying usability evaluations is the users' behavior. Additionally, they believe that their software does not have technical or usability problems. Other obstacles identified by novice developers were the difficulty in recruiting users/customers to participate in the tests, the design of the software, and finally, other technical and development organizations' issues.

The contribution helps to answer the second research question related to the resistance obstacle. Its findings confirm that, in the context where the main experiments of the PhD project were conducted, it is possible to identify several elements that shape the resistance obstacle.

3.3 Contribution 3

Lizano, F. & Stage, J. Improvement of novice software developers' understanding about usability: the role of empathy toward users as a case of emotional contagion. Accepted for publication in Proceedings of the 16th International Conference on Human-Computer Interaction (HCII) (2014)

This contribution reports the results of a quasi-experiment (William et al. 2002) conducted in a usability laboratory. The aim of the study was to make a preliminary exploration of the improvement of the understanding of usability and the empathy with users. The study considered the participation of computing students. The contribution presents the results of the study and compares these results with other studies.

The experiment included nine students grouped into two teams who conducted usability evaluations on software systems they had constructed. The usability evaluations were used to set up an environment where students could interact with users. The method used in the evaluations was the classical laboratory-based think-aloud protocol (Rubin, Chisnell 2008). During the study, two data collections related to the students' understanding of usability were conducted. The first data collection was made two weeks before the usability evaluations. The second was made immediately after the evaluations. Each data collection had two forms. The first form allowed participants to express their opinions related to the strengths and weaknesses of their software. The second form aimed at measuring the relative importance given by the participants to software/usability concepts in general. Additionally, several students were interviewed in order to elaborate on, or clarify, some findings of the study. The analysis focused on identifying the improvement in the understanding of usability by analyzing differences between the first and second data collections. The patterns of the understanding of usability after, and before, the data collection, were also analyzed by coding the opinions provided in the first form according to the taxonomy provided by Alonso-Ríos et al. (2009).

There are three main findings in this contribution. Firstly, after participation in the usability evaluations, participants improved their understanding of usability. Secondly, the analysis of the patterns after the usability evaluations indicated an increase in the number of students´ opinions that were more oriented to the users' needs – i.e., knowability, operability and subjective satisfaction (Alonso-Ríos et al. 2009). This fact suggests an improvement in the empathy towards users and their needs. Thirdly, the improvement in the students' empathy towards users was acquired in an unconscious process of emotional contagion (Rapson et al. 1993; Schoenewolf 1990; Barsade 2002) generated during the interaction with users.

This contribution helps in responding to the second research question because it provides a preliminary explanation of why those corrective actions which consider interaction with users (e.g., involving developers in usability evaluations) increase the understanding about usability and increase the empathy towards users. This improvement process can help to overcome the resistance obstacle.

3.4 Contribution 4

Lizano, F., Sandoval, M. M., & Stage, J. Integrating Usability Evaluations into Scrum: A Case Study Based on Remote Synchronous User Testing. Accepted for publication in Proceedings of the 16th International Conference on Human-Computer Interaction (HCII) (2014)

This contribution presents the results of an instrumental single case study (Lazar et al. 2010; Yin 2003) made in order to propose an example of how to integrate usability evaluations into a Scrum project. The case study had two detailed goals. Firstly, considering the Scrum iterative approach, it was necessary to explore which usability evaluation activities/artifacts should be used throughout the process in order to reduce the costs of the integration. Secondly, considering the widespread participation of software developers in the proposal of integration, it was necessary to explore the implications that such participation has in the integration – mainly in terms of how the developers' focus changes during the integration. This second goal was oriented to study the potential implications related to the resistance obstacle.

The case study was developed considering a software system designed to manage the data resulting from the supervision process of undergraduate students' projects. A case protocol (Yin 2003) was defined containing the case study specifications – e.g., period of time, location, hypothesis, etc. The data collection focused on the documents resulting from the tests and included the usability plans, the user tasks, the usability final reports and the guidelines used in the integration. The analysis used the general analytic strategy of relying on theoretical propositions (Yin 2003). Data evaluation focused on assessing usability reports delivered during the integration process, by using a checklist based on Capra's approach for describing usability problems (Capra 2006).

The integration approach presented in this contribution is based on conduction of usability evaluations by using the remote synchronous testing method (Andreasen et al. 2007). The integration considers nesting several usability tests into the Scrum's sprints scheme. Following the Hussain et al. (2012) approach for smaller usability evaluations, it is possible to conduct a smaller

usability evaluation in each sprint. Each test considers only two users performing a few usability tasks on specific parts of the software (normally 4–5 tasks per test). For each test, the users and the task must vary in order to achieve enough test coverage. In addition, the approach proposes the conduction of the usability tests by the software developers, as was suggested in previous studies (Bruun, Stage 2012; Skov, Stage 2012).

The proposed integration removes the necessity of using some of the usability evaluations' activities/artifacts due to the iterative scheme presented in the Scrum. For example, it is possible to define a single usability plan for all of the tests and to modify such a plan if required. In addition, by using remote synchronous testing, the participation of developers is practical and simple without spending time visiting a usability lab or the facilities where the users are working. The main disadvantage of the integration proposal is the problem developers have when they have to change their focus to conduct the usability evaluations. In the first test, made at the first sprint, the developers who participated in the usability evaluation forgot, or ignored, the rules documented in the guidelines. This problem was addressed by re-training efforts, which seemed to produce good results as the problem decreased.

This contribution helps to answer both research questions. In the case of the research question related to the cost, this contribution provides an integration approach that shows how it is possible to reduce costs in the usability evaluation process. In the case of the research question related to the resistance obstacle, the contribution warns that the developers' conduction of usability evaluations implies potential problems related to their natural focus on technical aspects.

3.5 Contribution 5

Lizano, F. & Stage, J. I see you: Increasing Empathy toward Users' Needs. In Proceedings of the 37th Information Systems Research Seminar in Scandinavia (IRIS 2014).

This contribution provides the results of a field study conducted to compare several usability evaluation methods. The aim of the study was to explore the efficiency and effectiveness of the methods in order to increase the software developers' understanding regarding usability and their empathy towards users. The study included two usability evaluation methods with users and two usability inspection methods.

The design of the study included a between-group design with four conditions corresponding to four methods of usability evaluation: the classical inspection method based on heuristic methods (Nielsen, Molich 1990), a variation of this inspection method with 'supervision', the classical laboratory-based think-aloud method (Rubin, Chisnell 2008), and a modern remote synchronous testing method (Andreasen et al. 2007). During the study, two data collections related to the participants' understanding of usability were conducted. In addition, several focus group sessions were conducted in order to validate and clarify the results. The analysis starts with a review process made in order to clarify the data collected. Later, all the information was coded to allow analysis of variations in the participants' understanding regarding usability before, and after, the usability evaluations. Statistics tests were used in order to identify significant differences in the data

collected. Finally, the data collected during the focus group sessions were quantified by using basic principles of the grounded theory (Strauss, Corbin 1998). This information was used to validate and clarify the findings of the study.

The findings indicate that those usability evaluation methods which consider interaction of developers with users can improve the developers' understanding and also increase their empathy towards users' needs. After the usability evaluations, findings reflect a change in the understanding regarding usability. This improvement was more evident in the participants who conducted the tests in the lab and by remote synchronous testing; these participants mentioned more aspects related to usability after the tests. In the case with lab conditions, there was an increase of 56% in the number of positive aspects, and 23% in the number of negative aspects. In the case with remote synchronous testing, there was an increase of 28% in the number of positive aspects, and 23% in the number of negative aspects. For both conditions, there was a significant difference in the number of positive aspects before, and after, the usability evaluations. This was not the case for the negative aspects, in which no significant difference was detected. This last situation can be explained by the developers' sense of individual ability and a personal identification with the software (Rasch, Tosi 1992; Hertel et al. 2003). Furthermore, comparing the number of positive or negative aspects reported in both conditions after the evaluations, there was no significant difference between those conducted in the lab and those by remote synchronous testing. This fact confirms similar effects of these methods on the participants' perspective. Conversely, the evaluations conducted by using the inspection methods had a lower impact in the change of the participants' understanding regarding usability. All these results were triangulated with the focus group sessions.

The contribution helps to answer the second research question related to the resistance obstacle. This contribution provides an empirical exploration of the effect of the use of several usability evaluation methods. The contribution allows us to conclude that through those usability evaluation methods which include interaction with users, it is possible to improve the understanding of usability and the empathy towards users' needs.

3.6 Contribution 6

Lizano, F. & Stage, J. Usability Evaluations for Everybody, Everywhere: A field study on Remote Synchronous Testing in Realistic Development Contexts. Accepted for publication in Proceedings of the 8th International Conference on Digital Society (ICDS) (2014)

This contribution reports the results of a field study made in order to compare two usability evaluation methods. The intention of the study was to explore how practical and cost effective each method was in allowing the software developers to conduct usability evaluations with users. The study included the remote synchronous test method and the classic laboratory-based think-aloud method.

The study considered several teams who conducted usability evaluations on software systems they had constructed. The software systems were the result of software development projects conducted in realistic contexts, close to practice (e.g., internal postal management system in a university,

laboratory equipment management in a biological research center, criminal records in a small municipal police station, students' records in a school, etc.). The scope of these projects was carefully controlled in order to guarantee similar characteristics in quality/size/complexity. The design of the study included a between-group design with two conditions corresponding to two methods of usability evaluation: the classical laboratory-based think-aloud method (Rubin, Chisnell 2008) and a modern remote synchronous testing method (Andreasen et al. 2007). In each condition, three usability evaluations were conducted on three different software systems. Each team evaluated another team's software. In the case of the laboratory method, the evaluations were conducted in a usability lab. In the remote synchronous testing method, the evaluations were made with the evaluators separated from users who remained in their jobs. The data collection was formed by the usability problem reports, the test logs and the videos collected during the tests. The data analysis starts by reviewing the consistency of the classification of the usability problems made by the participants. Due to the tests being conducted on different software systems, the analysis compared the differences between the two conditions by using average and standard deviations calculated separately for each condition. By using the test logs, it was possible to analyze the time spent in all the tests by calculating totals, averages and percentages.

The findings suggest that the remote synchronous test method allows the identification of a similar number of usability problems to those identified by the lab method. The number of usability problems identified under each of the conditions showed no significant difference. The task completion time confirms that the tests made by using the usability lab are more effective. Here, the users spent a total of 87.6 minutes completing the five tasks assigned to each one. The average time per user/task was 1.94 (SD=0.5). The average task completion time per usability problem identified under this method was 1.26. In the case of the remote synchronous method, the task completion time was 137.4, the average time per user/task was 3.10 (SD=1.3), and the average task completion time per problem was 2.32. Time spent conducting these tests was better using the remote synchronous method. The average time, in minutes, of tests using this method was lower (3290 minutes, SD=102) than the tests conducted in the lab (5910 minutes, SD=220.5). These results included all the actors involved in the tests (i.e., users, test monitor, logger, observers and technicians). The average time spent in the tests, for both methods, showed an extremely significant difference ($p<0.001$). By analyzing the time spent on each activity during the tests (i.e., preparation, conducting test, analysis and moving staff/users), it is possible to confirm significant differences for all of the activities.

The contribution helps to answer the first research question related to the cost obstacle. This contribution provides an empirical exploration of the effect of the use of two usability evaluation methods. The contribution allows us to conclude that the remote synchronous test method is more cost effective than the classical usability evaluations conducted in a lab.

4. Research Methods

In this chapter, I present the research methods used in my PhD project. In order to answer the research questions of my thesis, I used several research methods. Wynekoop and Conger (1992) proposed a classification of the main software engineering research into three major categories. The first category – environment independent setting – includes methods that allow the study of one or more phenomena without interaction with the research settings. The second category – artificial setting – considers those methods which allow the researcher to define particular settings in order to limit or control the experiment. The third category – natural setting – includes methods that study one or more phenomena in real organizations providing a strong empirical basis.

Based on this framework, in the following sub-sections I will present the methods used to answer the research questions. For each method I will present:

- A brief description of the method
- Main strengths and weaknesses
- How I have used each method
- How I attempted to overcome these weaknesses

In Table 3 it is possible see an overview of the research methods used in my PhD project.

Table 3. Research methods used in the PhD project.

Category	Method	RQ 1 (Cost)	RQ 2 (Resistance)
Environment Independent Setting	• Survey (contributions 1 and 2)	Help to understand facts related to the geographical context	Help to understand facts related to the participants involved in the research
Artificial Setting	• Laboratory Experiment – Quasi-experiment (contribution 3)		Identification of origin of improving of understanding of usability
Natural Setting	• Case Study (contribution 4)	Provide empirical basis for argumentation on integration of usability testing by RUT	Provide basis on empirical exploration of developers' potential behavioral issues.
	• Field Experiment (contributions 5 and 6)	Provide empirical basis for argumentation on the cost benefit of RUT	Provide empirical basis for argumentation on the handling of the developers' resistance

4.1 Survey

A survey is an environment independent setting method (Wynekoop, Conger 1992) oriented to collect data using different techniques such as mail or online questionnaires, phone interviews, forms, etc. After the data collection, it is possible to use statistical techniques in order to help in the interpretation of the main findings (Gable 1994). Surveys have several strengths and weaknesses. The main strengths are:

- Practical, economical alternative
- Helps to verify hypotheses
- Confidence in generalization

A survey is a practical and economical approach. This method allows the collection of data from a small or large number of participants by using diverse methods in a relatively unobtrusive way (Lazar et al. 2010). In addition, surveys help to verify hypotheses previously studied in particular contexts (Gable 1994; Attewell, Rule 1991). By conducting surveys in alternative contexts, and comparing results to previous studies, it is possible not only to verify hypotheses, but also to conduct a deeper exploration of general theories. Finally, the survey method provides a high level of confidence in the generalization of their results (Gable 1994; Jick 1979). This confidence in generalization is based on the use of representative samples which allow for the discovery of relationships between the variables considered in the survey. As a consequence of this generalization capability, surveys can increase predictability (Vidich, Shapiro 1955).

The main weaknesses of surveys are:

- Limited scope
- Difficult sampling
- Limited deep exploration

The scope of surveys is limited because this method helps to understand a particular phenomenon in a specific context and time (Gable 1994). In certain ways, these results represent a 'picture' that becomes outdated the moment it has ended. In addition, the sampling in the survey is one of the most debated matters surrounding this method (Scheuren 2004). The main discussion is how to define the size of the sample, what should be the composition of such a sample, and which participant selection method should be used. Finally, surveys avoid deep exploration (Lazar et al. 2010). The collection process of data in surveys is performed in a particular time, by using a specific technique with previously selected participants. Immediate interaction with participants in order to clarify, or extend, some preliminary findings is impossible.

In my PhD project, I conducted two surveys by using online questionnaires (contributions 1 and 2). One survey aimed to explore the application of usability evaluations in the software development organizations located in the geographical context of the project. The other survey aimed to explore the perspective of participants on usability evaluations.

Surveys have problems related to their limited scope, sampling and limited deep exploration. I have done the following to overcome these problems. Firstly, in the case of limitation in the scope, the results of my surveys were compared with other surveys made in different contexts and at different times. To facilitate this comparison, the design of my surveys considered the design of those previous studies. Secondly, as was expected, sampling became a challenge. Even considering that some strategies guaranteeing the quality of the sample were done (e.g., filtering participants, randomizing the selection of participants, etc.), I would have wished to apply, in a different way, other elements of the sampling process such as, for example, having access to a wider variety of participants. Finally, the deep exploration issue was not a problem for me, because the results of the surveys were not only compared with previous studies, but were also used as a complement for other studies' results.

I can only generalize within these limitations; I have learned that in the sampling process it is necessary to carefully consider the different elements of the process; quality of the sampling's results may be compromised even if only a small part of the whole process is weak.

4.2 Laboratory experiment

In my PhD project I used a laboratory experiment by conducting a quasi-experiment. Wynekoop and Conger (1992) classified the laboratory experiment method as an artificial setting method, which is ideal in controlled experiments of theory or product testing. Meanwhile, William et al. (2002) defined a quasi-experiment as a research method in which units are not assigned randomly. In Table 4 it is possible to observe the main strengths and weaknesses of the laboratory and quasi-experiment methods.

Table 4. Main strengths and weaknesses of the laboratory and quasi-experiment methods.

Method	Strengths	Weaknesses
Laboratory	• Control • Helps identify relationships	• Lack of contextualization • Unknown level of generalization
Quasi-experiment	• Allows similar formalism • Practical, economical alternative	• Limited hypothetical inference • Limited validation

Laboratory experiments allow the researcher to focus on specific phenomena with full control of the research variables (Wynekoop, Conger 1992). In addition, this method enables precise identification of relationships between chosen variables by using quantitative analytical techniques (Braa, Vidgen 1999). The main weaknesses of laboratory experiment methods are related to their lack of contextualization as a result of their limited relation to real contexts. This problem produced the second main weakness, which is the unknown level of generalization of the outcomes of the laboratory experiment (Kjeldskov, Graham 2003).

Meanwhile, the main strengths associated with the quasi-experiment are that, as a research method, it shares similar purposes and structural details with other more formal research methods (William et al. 2002). Additionally, this method is a practical and economical alternative in those cases where circumstances prevent a more real experiment (Easterbrook et al. 2008; Ross, Morrison

1996). However, a quasi-experiment also has weaknesses such as the limited hypothetical inference due to the lack of randomization (William et al. 2002), and the limited validation of results due to the same lack of randomization (Ross, Morrison 1996).

The motivation for using a quasi-experiment in my PhD project (contribution 3) was to provide preliminary argumentation to help in response to the research question related to the software developers' resistance to accepting users' opinions. Considering that some studies suggest an improvement of developers' understanding regarding usability by integrating them into usability evaluation activities (Skov, Stage 2012; Hoegh et al. 2006), I was interested in identifying the possible origin of such a phenomenon, and of even greater relevance, to better understand the relation between the improvement of the understanding and the empathy towards users' needs.

The laboratory method has problems related to the lack of contextualization and an unknown level of generalization. To overcome this problem, I only considered software engineering and computer science students as participants, as they share similar characteristics to the novice software developers (Bruun, Stage 2012). In addition, the quasi-experiment's main problems are the limitations in its hypothetical inference and with its validation. I have done the following to overcome these problems. Firstly, to control the hypothetical inference, I selected only participants who share similar curricula and demographic characteristics. All the students were male, were of similar age, had studied on the same courses and had shared interests. Secondly, validation of the quasi-experiment's results was made by contrasting these results against other studies in order to identify facts that support the results. This was the case of the unconscious process of emotional contagion presented in the experiment.

The unknown level of generalization of the results of my experiment persists as an undeniable fact. The main lesson learned is related to improving the design of the experiment in order to consider more realistic contexts which guarantee a higher level of generalization.

4.3 Case study

Wynekoop and Conger (1992) have classified case studies as a natural setting method. This is a method focused on contemporary phenomena with some context of real application in daily life. The main uses of the method are: to describe and explain a phenomenon or situation, to develop hypothesis, and to respond to "how" or "why" questions (Yin 2003).

As a research method, a case study has several strengths and weaknesses. The main strengths are:

- Contextual versatility
- Allows deeper research

A case study is a versatile research method. It helps to increase knowledge about phenomena in diverse contexts (e.g., individual, group, organizational, social, political, etc.). As a research strategy, it is widely used in psychology, sociology, political science, social work, business and community planning. There are many research aims where case studies are useful. In information systems research, case studies are especially useful because of their numerous advantages (Benbasat, 1984).

For example, the researcher can study systems in a natural setting and propose new theories which consider not only observation of such systems but also the state of the art.

Additionally, a case study allows deeper exploration of certain areas which lack previous studies (Cavaye 1996), and this is something that allows development or testing of existing theories. Coincidentally, Darke (1998) argues that this method allows not only deeper exploration of existing theories but also provides new descriptions of phenomena as well as developing entirely new theories.

A case study's main weaknesses are:

- Subjectivity
- Poor generalization

Subjectivity is present in case studies due to the intervention of the researcher in defining specific data collections and particular analysis processes (Darke 1998). This subjectivity is more evident considering the characteristics of versatility and deeper research that the case study method allows.

Poor generalization is a relevant criticism made of the case study approach as a consequence of focusing on a particular phenomenon under study (Abercrombie et al. 1984). Critics of the method argue that the limitation of a single case under study can undermine efforts to generalize results to a larger scope.

I conducted an instrumental single case study (contribution 4) in order to develop an example of how to integrate RUT into a Scrum software project. I was interested in exploring the context of the integration, the cost and the developers' behavior patterns.

The case study method has problems related to subjectivity and poor generalization. The subjectivity issue was handled by using the 'relying on theoretical propositions' analysis approach suggested by Yin (2003). This analysis approach guided me in the analysis process by avoiding, as much as possible, personal inferences about how to conduct such a process. This approach recommends conducting the analysis by using a framework formed by the theory used in the design of the study, research questions, literature, etc. The theory used in my case study was related to the known efforts of integration of usability evaluations into software projects. This theory has oriented me in the design and also in the formulation of the research question. Additionally, the analysis of the results of the case study allowed me to confirm many of the findings of previous studies and also generate new contributions to the integration theory.

In the case of the limited generalization, even considering the effectiveness of the analysis approach suggested by Yin (2003), the small size of the case study certainly may limit a strong generalization.

I can only generalize within these limitations. The main lesson learned in the use of this method is the importance of having alternatives to generate more confidence in the results of the case study, thus increasing generalization. In the specific case of the size of the case study, although the case

study has incorporated important methodological elements, and also the main theory of integration, its limited size can imply risk to such generalization.

4.4 Field experiment

Wynekoop and Conger (1992) have classified the field experiment as a natural setting method normally used for studying current practice and for evaluating new practices. As with any research method, the field experiment has strengths and weaknesses. The main strengths are:

- Practical
- Realistic settings

Braa and Vidgen (1999) argue that the field experiment method is an extension of lab experiments conducted in the particular context of an organization, and this is something that implies less methodological rigor but conduction in a more realistic environment. The realistic settings used in a field experiment are useful in terms of exploring a specific phenomenon in conditions close to reality. For example, observation of users' natural behaviors in their own environments was highlighted by Nielsen (2012a) as an important method used in HCI research.

The main weaknesses of the field experiment are:

- Difficult to find an adequate setting
- Control and management

Considering that to have an adequate environment is a key aspect of the design of the field experiment, the difficulty in finding such an environment becomes a relevant weakness (Braa, Vidgen 1999). Real environments may limit the research process – e.g., time restrictions, resource limitations, motivation of participants, etc. Another weakness is the complexity of the control and management process (Kjeldskov, Graham 2003). The particular characteristics of the field experiment (i.e., made outside of controlled conditions existing in a lab) make the process complex. For example, a variety of logistics must be considered in the experimental design, as well as the particular conditions presented in the place where the experiment will be conducted. The management process of, among other things, the data collection is also complex and demands additional efforts considering that the experiment setting is not necessarily pre-conditioned to allow conduction of regular experiments.

I conducted two field experiments in order to compare several usability evaluation methods. The first field experiment aimed to explore the efficiency and effectiveness of the methods in increasing the software developers' understanding of usability and their empathy towards users. The second field experiment aimed to explore how practical and cost effective the methods were for allowing the software developers to conduct usability evaluations with users.

Both field experiments (contributions 5 and 6) had the same problems: finding adequate environments, and controlling and managing the process.

To overcome the difficulty of finding an adequate environment, I did the following. First, I defined a set of conditions that the potential organizations and participants had to meet. Second, once I had identified potential actors, I randomly selected the number of organizations and participants needed for the experiments. Finally, again by using a random distribution, I grouped the actors into the different conditions used in both experiments.

To overcome the problem of control and management, I did the following. First, I defined several guidelines to orientate the conduction of the experiment. Second, I provided formal training to the experiment's participants. Third, I provided personalized advice to the participants by using alternative channels (i.e., in person, email, chat and phone). Finally, all the data collections were backed up by using different alternatives (e.g., CD-ROM copies, public file hosting services, public video-sharing websites, etc.). Although all these measures were taken, it is a fact that the public nature of some tools used to back up the data collection (i.e., the hosting services and video-sharing websites), involves a certain level of risk for such data collections.

I can only generalize within these limitations; I have learned that in the control and management processes of the field experiments, it is necessary to consider a comprehensive set of actions from the beginning of the process until the final backup of the data. These actions include formal and secure backup of the data collections. Potential risks to the data collections can seriously compromise the generalization of the experiments' results.

4.5 General limitation in the research methods

I did not have the opportunity to include more experienced software developers in my PhD project. As an alternative, advanced computing students were included in the surveys and the series of empirical studies. These advanced students were considered as novice software developers because they share similar characteristics. I base such a statement on three main facts.

Firstly, Bruun and Stage (2012) defined novice developers as persons with limited job experience related to usability engineering and no formal training in usability engineering methods. In this sense, advanced students share similar characteristics to the novice developers.

Secondly, the students mainly conducted usability evaluations in my PhD project. To perform these activities, students had to use several soft skills – e.g., defining users' tasks, documenting the results, following a method, working with real users, working in teams, etc. According to Begel and Simon (2008), novice developers (as well as the students who participated in my research), usually have serious constraints when it comes to these soft skills because these issues are normally less well supported in university pedagogy.

Finally, as with novice developers, the students who participated in my PhD project were not preconditioned with extensive previous work experience.

5. Conclusions

In this chapter, I present the conclusions of the thesis by providing conclusions on the research questions, limitations in the research process, and suggestions for future work.

5.1 Research question 1

Research question 1 is: How can software developers use remote usability testing to conduct usability evaluations in an economical way?

To answer this research question, I conducted several studies including an environment independent setting (survey reported in contribution 1) and two natural setting studies (case study reported in contribution 4 and field experiment reported in contribution 6).

First, through the survey reported in contribution 1, it is possible to confirm that, in the context used in my PhD project, the 'perceived cost obstacle' exists in a similar form to that reported in other studies (Ardito et al. 2011; Bak et al. 2008). By using alternative ways (i.e., open and closed questions), the survey clearly identified the existing perception in development organizations related to the cost of usability evaluations, and how this is an important obstacle in applying such tests. Confirming this fact, in the context of my PhD, was important because it provided me with the confidence that the results of the other empirical studies conducted in my PhD project can be compared to other studies made in alternative contexts, places and times.

Second, having confirmed the existence of the 'perceived cost obstacle' in the context of my research, I proposed an integration approach of usability evaluation into a Scrum project (contribution 4). Contribution 4 shows the feasibility of integration in an economical way. One of the most relevant elements of the integration is the participation of software developers conducting the usability evaluations (Bruun, Stage 2012; Skov, Stage 2012). By using developers to conduct usability evaluations, it was not necessary to hire external independent usability experts, thus reducing the cost of the process as suggested by Bruun (2013). In addition, the integration approach considers the use of a specific RUT method, called remote synchronous testing (Andreasen et al. 2007), and other recommendations previously proposed in the literature – i.e., small usability evaluations (Hussain et al. 2012), the use of an iterative scheme used in agile methods (Sohaib, Khan 2010) and the use of SE terminology based on international standards (Fischer 2012). All of these elements allow an economical integration that is more 'natural' in terms of an agile method, such as Scrum, allowing developers to respond rapidly to the changes in a software development process in order to reduce risks and costs (Beck, Andres 2004).

Finally, through a field experiment (contribution 6), I have confirmed that remote synchronous testing is a practical and cost-effective alternative for integrating usability evaluations into software projects. My experiment indicated that there is no statistical significant difference in the number of problems identified by using this RUT method when compared to the usability lab. Even considering that the task completion time in the lab was 37% quicker, the time spent to complete all of the remote synchronous tests was 44% quicker. The statistical analysis has shown that the difference was extremely significant. These results included all the actors involved in the tests (i.e., users, test

monitor, logger, observers, etc.), which implies a more real context in terms of the whole testing process. In this case, the field experiment has shown that with RUT it is possible to reduce the 'actual cost obstacle' in order to allow economical conduction of usability evaluations.

In conclusion, software developers can use RUT, specifically the remote synchronous testing method, to conduct complete usability evaluations in 44% less time, obtaining similar results to those obtained in the usability lab. The software developers can integrate usability evaluations into modern software development processes as for example in those development processes conducted by using agile methods. To do this, developers can use an iterative scheme of small tests by using the remote synchronous testing method.

5.2 Research question 2

Research question 2 is: How can remote usability testing reduce the resistance from software developers and make them accept users' opinions about the usability of a software system?

To answer this research question, I conducted four studies: an environment independent study (survey reported in contribution 2), an artificial setting study (quasi-experiment in the lab reported in contribution 3) and two natural setting studies (case study reported in contribution 4 and field experiment reported in contribution 5).

The first study was a survey reported in contribution 2. This survey allowed me to confirm that it is possible to observe some elements of the resistance obstacle through the participants in my PhD project. For example, the preference for software technical aspects and the belief that users are the main obstacle in applying usability evaluations are indicators of what Ardito et al. (2011) and Bak et al. (2008) called the software developers' mindset. This fact confirms a certain lack of understanding presented by the participants (Bak et al. 2008; Rosenbaum et al. 2000; Seffah, Metzker 2004); even considering that the survey indicates that the participants had a relatively good understanding of what the usability evaluations are, the survey's results also allow an inference that their understanding of the users and their needs are not clear. This confirmation provided me with the confidence that the results of my PhD project can be compared to other results in order to respond to research question 2.

The second study was a quasi-experiment in the usability lab (contribution 3). This experiment allowed me to understand the origin of the improvement in the developers' understanding of usability (Hoegh et al. 2006; Skov, Stage 2012). My experiment revealed that, during the usability evaluations, developers do not necessarily focus on the users. In such moments, they concentrate more on the software and its problems. However, at the same time there is an unconscious contagion of developers by the users' emotions. This finding confirms previous studies, which argued that by participating in, or observing, usability evaluations, developers improve their understanding and empathy (Hoegh et al. 2006; Skov, Stage 2012; Gilmore, Velázquez 2000). In addition, this experiment confirms the existence of an emotional contagion process when developers see users working with a software system in the context of a usability evaluation (Rapson et al. 1993; Schoenewolf 1990; Barsade 2002). This unconscious contagion process, which

precedes the increase in empathy (De Vignemont 2004; Singer, Lamm 2009), explains why empathy increases during the usability evaluation.

The third study was a case study reported in contribution 4. This case study exposed an example of how it is possible to integrate the usability evaluations into a software development process. The case study confirmed that this approach implies problems for the developers, specifically when they need to change their roles as developers in order to conduct usability evaluations.

Finally, the last study was a field experiment (contribution 5). Findings of this study complement the results presented in contribution 3. In the field experiment, I confirmed that remote synchronous testing has a similar effectiveness to the usability lab in improving the developers' understanding regarding usability. In parallel, the remote synchronous testing method also increases developers' empathy towards users. This increase of developers' understanding and empathy confirms, in more real contexts, previous studies regarding the benefit of involving developers in usability evaluation activities (Skov, Stage 2012; Hoegh et al. 2006). In addition, the field experiment confirmed the unconscious contagion of emotions reported in contribution 3 and other studies (De Vignemont 2004; Singer, Lamm 2009). Finally, the field experiment confirmed that using remote synchronous testing allows a 'remote contagion of emotions' (Hancock et al. 2008; Kramer 2012). This fact is relevant because it justifies the use of this method to take advantage of the emotional contagion process, even if the observer and observed are not face to face.

In conclusion, remote synchronous testing can reduce the resistance of software developers to accepting users' opinions by setting up an environment in which developers can interact with users in a usability evaluation context. This approach improves the developers' understanding regarding usability and enables the unconscious contagion process of users' emotions – something that will increase the developers' empathy towards users.

5.3 Overall research question

The overall research question is: How can remote usability testing contribute to resolving the cost and resistance obstacles by providing an effective and efficient integration of usability evaluations into a software development process?

A synchronous RUT method, such as remote synchronous testing, contributes to resolving both obstacles.

Firstly, the participation of software developers in usability evaluations is a key element of the strategy for resolving the resistance obstacle. RUT methods are effective because they allow, in a practical way, the participation of software developers in usability evaluations. With this participation, RUT sets up an environment in which developers can interact with users. This interaction helps to improve the software developers' understanding and, more importantly, helps to increase the developers' empathy towards users and their needs. The effectiveness of RUT is grounded in the fact that it allows a remote interaction with users. This remote interaction is as effective as the face-to-face interaction present at the usability lab. In addition, effectiveness of RUT

is also related to the fact that, by reducing the resistance obstacle, the integration is more natural; the developers are not only able to improve their understanding of, and empathy towards, users' needs, they can also learn about the process in order to use it in the future.

Secondly, RUT methods can help to resolve the 'cost obstacle'. RUT methods, such as remote synchronous testing, allows the obtaining, in a cost-efficient way, of similar results to those at the usability lab. In addition, the virtualization of the process also saves resources by avoiding unnecessary movement of staff or users to conduct usability evaluations. The efficiency of RUT relies on the fact that all the activities in the test process require much less time than at the lab, while still obtaining similar results. In addition, the efficiency of RUT is also related to the fact that the actual cost of the usability evaluations is lower than at the lab. Because it is easier to justify usability evaluations, integration becomes more feasible.

In conclusion, RUT sets up a cost-efficient environment in which software developers can effectively interact with users in order to reduce the resistance obstacle.

5.4 Limitations of the research

In my PhD Project, I used several research methods. The limitations of the entire research are intrinsically related to the constraints presented in each method. Even though I have employed some countermeasures, some limitations were not possible to control.

In the case of the research methods related to research question 1 (i.e., cost), there were limitations in one survey (contribution 1), the case study (contribution 4) and one field experiment (contribution 6).

In the case of the survey, the main limitation was related to the number of participants. There is a permanent debate surrounding this issue (Albert, Tullis 2013; Scheuren 2004; Lazar et al. 2010). The decision regarding the size of samples considers different factors. For example, Albert and Tullis (2013) argue that, in the sampling size, it is necessary to consider the diversity of user population, complexity of the product and the specific aims of the study. Lazar et al. (2010) consider that this matter depends on the level of confidence and which margin of error is considered acceptable. Finally, Scheuren (2004) says that such a decision depends on financial resources available for the study.

For the case study, the main limitation was its small size, which may limit the confidence in generalization. A limited sized case study has associated risks in trying to generalize results, of an idiosyncratic group of participants, to others (Lazar et al. 2010). Yin (2003) agrees with this viewpoint but, at the same time, argues that this is quite a universal problem present in experiments. Moreover, Yin (2003) believes that generalization in science commonly needs more than one experiment or condition.

Finally, for the field experiment, the main limitation was related to a certain level of risk in managing the data collections due to the use of relatively unsecure tools. The use of software tools to manage

data collections has increased in HCI experiments (Lazar et al. 2010). The public file hosting services and public video-sharing websites are especially interesting due to the resource savings they represent. Consequently, it will always be necessary to look for secure ways of using such economical tools.

In the case of the research methods related to research question 2 (i.e., resistance), the main limitations were located in one survey (contribution 2), the quasi-experiment made in the usability laboratory (contribution 3), the case study (contribution 4) and one field experiment (contribution 5). For the majority of cases, the limitations and the countermeasures were the same as those discussed above. In the quasi-experiment conducted in the usability laboratory, the unknown level of generalization for this kind of research method persists as an undeniable fact (William et al. 2002). Even considering that I took some actions to increase the confidence in generalization (e.g., contrasting results with other studies, selecting participants with similar characteristics, etc.), the same nature of the quasi-experiment constrains the possibility of having a clear idea of the generalization level.

5.5 Future work

This PhD project is a foundation for continued research in at least three different ways.

Firstly, this research was focused on one RUT method: remote synchronous testing. There are different RUT methods (Andreasen et al. 2007). An interesting future research line could be to explore the efficiency and effectiveness of the integration approach suggested in this thesis, by using other RUT methods – especially some asynchronous methods. It could be interesting to explore how these methods overcome the cost obstacle, but it would be more interesting to explore if these asynchronous methods can overcome the resistance obstacle.

Secondly, this PhD project exposed an alternative application of the emotional contagion theory. Considering that my research can be mainly generalized to novice software developers, it is necessary to continue the research in order to explore how the main concepts of the emotional contagion theory interact with other kinds of software developers. For example, it is necessary to explore the results of contagion processes in those cases of more experienced developers who have different value judgments, or good/bad experiences, related to usability. Similarly, it is interesting to explore the role of pressure in groups; for example, how the contagion process works in situations where different groups, with strong and entrenched positions, should share emotions related to users' needs or usability in general.

Finally, even considering the fact that throughout the entire PhD project, I conducted surveys and a series of empirical studies in contexts close to practice, it is necessary to continue with additional longitudinal studies. These studies will help to reinforce the results obtained in this investigation and can be compared to usability evaluation methods that are commonly used in software development practice.

References

Abercrombie, N., Hill, S., & Turner, B. S. (1984). *Dictionary of sociology*. Penguin Books.

Abran, A., Moore, J. W., Bourque, P., Dupuis, R., & Tripp, L. L. (2004). Guide to the Software Engineering Body of Knowledge: 2004 Edition-SWEBOK. *IEEE Computer Society*.

Airaksinen, T., & Byström, E. E. (2007). User and Business Value: A Dual-Stakeholder Perspective on IT Systems.

Albert, W., & Tullis, T. (2013). Measuring the user experience: collecting, analyzing, and presenting usability metrics. Newnes.

Allen, B. (1996). *Information tasks: Toward a user-centered approach to information systems*. Academic Press, Inc..

Alonso-Ríos, D., Vázquez-García, A., Mosqueira-Rey, E., & Moret-Bonillo, V. (2009). Usability: a critical analysis and a taxonomy. International Journal of Human-Computer Interaction, 26(1), 53-74

Alshaali, S. (2011). *Human-computer interaction: lessons from theory and practice* (Doctoral dissertation, University of Southampton).

Andreasen, M., Nielsen, H., Schrøder, S., & Stage, J. (2006). Usability in open source software development: opinions and practice. *Information technology and control*, *25*(3A), 303-312.

Andreasen, M. S., Nielsen, H. V., Schrøder, S. O., & Stage, J. (2007, April). What happened to remote usability testing?: an empirical study of three methods. In *Proceedings of the SIGCHI conference on Human factors in computing systems* (pp. 1405-1414). ACM.

Ardito, C., Buono, P., Caivano, D., Costabile, M. F., Lanzilotti, R., Bruun, A., & Stage, J. (2011). Usability evaluation: a survey of software development organizations. In *SEKE* (pp. 282-287).

Attewell, P., & Rule, J. B. (1991). Survey and other methodologies applied to IT impact research: experiences from a comparative study of business computing.*The Information systems research challenge: survey research methods*, *3*, 299-315.

Bak, J. O., Nguyen, K., Risgaard, P., & Stage, J. (2008, October). Obstacles to usability evaluation in practice: a survey of software development organizations. In *Proceedings of the 5th Nordic conference on Human-computer interaction: building bridges* (pp. 23-32). ACM.

Barsade, S. G. (2002). The ripple effect: Emotional contagion and its influence on group behavior. *Administrative Science Quarterly*, *47*(4), 644-675.

Bartek, V., & Cheatham, D. (2003). Experience remote usability testing, Part 1.*IBM Developer Works*.

Beck, K., & Andres, C. (2004). *Extreme programming explained: embrace change*. Addison-Wesley Professional.

Beck, K., Beedle, M., Van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., ... & Thomas, D. (2001). Principles behind the agile manifesto.*Agile Alliance*.

Begel, A., & Simon, B. (2008, September). Novice software developers, all over again. In *Proceedings of the Fourth international Workshop on Computing Education Research* (pp. 3-14). ACM.

Bellotti, V. (1988, October). Implications of current design practice for the use of HCI techniques. In *Proceedings of the Fourth Conference of the British Computer Society on People and computers IV* (pp. 13-34). Cambridge University Press.

Benbasat, I. (1984). An analysis of research methodologies. *The information systems research challenge*, 47-85.

Bolt, N., Tulathimutte, T., & Merholz, P. (2010). *Remote research*. New York: Rosenfeld Media.

Borgholm, T., & Madsen, K. H. (1999). Cooperative usability practices.*Communications of the ACM*, *42*(5), 91-97.

Bourque, P., Fairley, R. (2014). SWEBOK : Guide to the Software Engineering Body of Knowledge Version 3.0. *IEEE Computer Society*.

Braa, K., & Vidgen, R. (1999). Interpretation, intervention, and reduction in the organizational laboratory: a framework for in-context information system research. *Accounting, Management and Information Technologies*, *9*(1), 25-47.

Brereton, E. (2005). Don't neglect usability in the total cost of ownership.*Communications of the ACM*, *47*(7), 10-11.

Brooke, J. (1996). SUS-A quick and dirty usability scale. *Usability evaluation in industry*, *189*, 194.

Bruun, A., Gull, P., Hofmeister, L., & Stage, J. (2009, April). Let your users do the testing: a comparison of three remote asynchronous usability testing methods. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 1619-1628). ACM.

Bruun, A., & Stage, J. (2012, November). Training software development practitioners in usability testing: an assessment acceptance and prioritization. In *Proceedings of the 24th Australian Computer-Human Interaction Conference*(pp. 52-60). ACM.

Bruun, A. (2013). *Developer Driven and User Driven Usability Evaluations*(Doctoral dissertation, Videnbasen for Aalborg UniversitetVBN, Aalborg UniversitetAalborg University, Det Teknisk-Naturvidenskabelige FakultetThe Faculty of Engineering and Science).

BugHuntress, Q. A. (2007). Lab, Mobile Usability Testing Problem and Solutions. In *Proceedings of the Conference Quality Assurance: Management & Technologies (QAMT Ukraine'07)*.

Capra, M. G. (2006). *Usability problem description and the evaluator effect in usability testing* (Doctoral dissertation, Virginia Polytechnic Institute and State University).

Cavaye, A. L. (1996). Case study research: a multi-faceted research approach for IS. Information systems journal, 6(3), 227-242.

Dandavate, U., Sanders, E. B. N., & Stuart, S. (1996, October). Emotions matter: user empathy in the product development process. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* (Vol. 40, No. 7, pp. 415-418). SAGE Publications.

De Vignemont, F. (2004). The co-consciousness hypothesis. *Phenomenology and the Cognitive Sciences*, *3*(1), 97-114.

Decety, J., & Jackson, P. L. (2006). A social-neuroscience perspective on empathy. *Current directions in psychological science*, *15*(2), 54-58.

Dray, S., & Siegel, D. (2004). Remote possibilities?: international usability testing at a distance. *interactions*, *11*(2), 10-17.

Darke, P., Shanks, G., & Broadbent, M. (1998). Successfully completing case study research: combining rigour, relevance and pragmatism. Information systems journal, 8(4), 273-289.

Easterbrook, S., Singer, J., Storey, M. A., & Damian, D. (2008). Selecting empirical methods for software engineering research. In Guide to advanced empirical software engineering (pp. 285-311). Springer London.

Ehrlich, K., & Rohn, J. (1994). Cost justification of usability engineering: A vendor's perspective. *Cost-justifying usability*, 73-110.

Ferré, X., Juristo, N., Windl, H., & Constantine, L. (2001). Usability basics for software developers. *IEEE software*, *18*(1), 22-29.

Ferre, X., Juristo, N., & Moreno, A. M. (2005). Framework for integrating usability practices into the software process. In *Product focused software process improvement* (pp. 202-215). Springer Berlin Heidelberg.

Ferré, X., Juristo, N., & Moreno, A. M. (2006). Obstacles for the integration of hci practices into software engineering development processes. *Encyclopedia of HCI*, 422-442.

Fidgeon, T., (2011). Usability Testing: When to use remote usability testing. Available from: http://www.spotlessinteractive.com/articles/usability-research/usability-testing/remote-usability-testing-when-to-use.php [Accessed 3 January 2014]

Fischer, H. (2012, June). Integrating usability engineering in the software development lifecycle based on international standards. In *Proceedings of the 4th ACM SIGCHI symposium on Engineering interactive computing systems*(pp. 321-324). ACM.

Fulton Suri, J. (2003). Empathic design: Informed and inspired by other people's experience. *Empathic Design-User experience in product design*, 51-57.

Gable, G. G. (1994). Integrating case study and survey research methods: an example in information systems. *European Journal of Information Systems*,*3*(2), 112-126.

Gilmore, D. J., & Velázquez, V. L. (2000, April). Design in harmony with human life. In *CHI'00 Extended Abstracts on Human Factors in Computing Systems*(pp. 235-236). ACM.

Göransson, B., Gulliksen, J., & Boivie, I. (2003). The usability design process–integrating user-centered systems design in the software development process.*Software Process: Improvement and Practice*, *8*(2), 111-131.

Granollers, T., Lorés, J., & Perdrix, F. (2003). Usability engineering process model. Integration with software engineering, *HCI-Intl'03*, Crete-Greece

Grudin, J. (1991). Obstacles to user involvement in software product development, with implications for CSCW. *International Journal of Man-Machine Studies*, *34*(3), 435-452.

Gulliksen, J., Boivie, I., Persson, J., Hektor, A., & Herulf, L. (2004, October). Making a difference: a survey of the usability profession in Sweden. In*Proceedings of the third Nordic conference on Human-computer interaction* (pp. 207-215). ACM.

Hancock, J. T., Gee, K., Ciaccio, K., & Lin, J. M. H. (2008, November). I'm sad you're sad: emotional contagion in CMC. In *Proceedings of the 2008 ACM conference on Computer supported cooperative work* (pp. 295-298). ACM.

Hartson, H. R., Castillo, J. C., Kelso, J., & Neale, W. C. (1996, April). Remote evaluation: the network as an extension of the usability laboratory. In*Proceedings of the SIGCHI conference on Human factors in computing systems* (pp. 228-235). ACM.

Hartson, H. R., & Castillo, J. C. (1998, May). Remote evaluation for post-deployment usability improvement. In *Proceedings of the working conference on Advanced visual interfaces* (pp. 22-29). ACM.

Hertel, G., Niedner, S., & Herrmann, S. (2003). Motivation of software developers in Open Source projects: an Internet-based survey of contributors to the Linux kernel. *Research policy*, *32*(7), 1159-1177.

Hoegh, R. T., Nielsen, C. M., Overgaard, M., Pedersen, M. B., & Stage, J. (2006). The impact of usability reports and user test observations on developers' understanding of usability data: An exploratory study. *International journal of human-computer interaction*, *21*(2), 173-196.

Holzinger, A. (2005). Usability engineering methods for software developers.*Communications of the ACM*, *48*(1), 71-74.

Hussain, Z., Lechner, M., Milchrahm, H., Shahzad, S., Slany, W., Umgeher, M., ... & Wolkerstorfer, P. (2012, January). Practical Usability in XP Software Development Processes. In *ACHI 2012, The Fifth International Conference on Advances in Computer-Human Interactions* (pp. 208-217).

Hvannberg, E. T., & Law, E. L. C. (2003). Classification of Usability Problems (CUP) Scheme. In INTERACT.

Jeffries, R., Miller, J. R., Wharton, C., & Uyeda, K. (1991, March). User interface evaluation in the real world: a comparison of four techniques. In*Proceedings of the SIGCHI conference on Human factors in computing systems* (pp. 119-124). ACM.

Jerome, B., & Kazman, R. (2005). Surveying the solitudes: An investigation into the relationships between human computer interaction and software engineering in practice. In *Human-Centered Software Engineering—Integrating Usability in the Software Development Lifecycle* (pp. 59-70). Springer Netherlands.

Jia, Y. (2012) Examining Usability Activities in Scrum Projects–A Survey Study (Doctoral dissertation, Uppsala University).

Jick, T. D. (1979). Mixing qualitative and quantitative methods: Triangulation in action. *Administrative science quarterly*, 602-611.

Juristo, N., & Ferre, X. (2006, May). How to integrate usability into the software development process. In *Proceedings of the 28th international conference on Software engineering* (pp. 1079-1080). ACM.

Kantner, L., & Rosenbaum, S. (1997, October). Usability studies of WWW sites: heuristic evaluation vs. laboratory testing. In *Proceedings of the 15th annual international conference on Computer documentation* (pp. 153-160). ACM.

Kjeldskov, J., & Graham, C. (2003). A review of mobile HCI research methods. In *Human-computer interaction with mobile devices and services* (pp. 317-335). Springer Berlin Heidelberg.

Kjeldskov, J., Skov, M. B., & Stage, J. (2004, October). Instant data analysis: conducting usability evaluations in a day. In *Proceedings of the third Nordic conference on Human-computer interaction* (pp. 233-240). ACM.

Kramer, A. D. (2012, May). The spread of emotion via facebook. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 767-770). ACM.

Lazar, J., Feng, J. H., & Hochheiser, H. (2010). *Research methods in human-computer interaction*. John Wiley & Sons.

Lindgaard, G., & Chattratichart, J. (2007, April). Usability testing: what have we overlooked?. In *Proceedings of the SIGCHI conference on Human factors in computing systems* (pp. 1415-1424). ACM.

Mattelmäki, T., & Battarbee, K. (2002, January). Empathy probes. In *PDC* (pp. 266-271).

Meiselwitz, G., Wentz, B., & Lazar, J. (2010). *Universal usability: Past, present, and future*. Now Publishers Inc.

Menghini, F., (2006). Remote usability testing. Available from: http://internotredici.com/article/remoteusabilitytesting/ [Accessed 3 January 2014]

Muzaffar, A. W., Azam, F., Anwar, H., & Khan, A. S. (2011). Usability Aspects in Pervasive Computing: Needs and Challenges. *International Journal of Computer Applications*, *32*.

Newell, A. F., Morgan, M. E., Gregor, P., & Carmichael, A. (2006, April). Theatre as an intermediary between users and CHI designers. In *CHI'06 Extended Abstracts on Human Factors in Computing Systems* (pp. 111-116). ACM.

Nichols, D., & Twidale, M. (2003). The usability of open source software. *First Monday*, *8*(1).

Nielsen, J. (1993) *Usability Engineering*. Morgan Kaufmann Publishers Inc.

Nielsen, J. (1994). Guerrilla HCI: Using discount usability engineering to penetrate the intimidation barrier. *Cost-justifying usability*, 245-272.

Nielsen, J. (2012a). Best Application Designs. http://www.nngroup.com/articles/best-application-designs/ [Accessed 3 January 2014]

Nielsen, J. (2012b). Thinking aloud: the # 1 usability tool. URL: http://www. nngroup. com/articles/thinking-aloud-the-1-usability-tool [Accessed 3 January 2014]

Nielsen, J., & Molich, R. (1990, March). Heuristic evaluation of user interfaces. In *Proceedings of the SIGCHI conference on Human factors in computing systems* (pp. 249-256). ACM.

Paternò, F. (2003, November). Models for universal usability. In *Proceedings of the 15th French-speaking conference on human-computer interaction on 15eme Conference Francophone sur l'Interaction Homme-Machine* (pp. 9-16). ACM.

Patton, J. (2002, November). Hitting the target: adding interaction design to agile software development. In *OOPSLA 2002 Practitioners Reports* (pp. 1-ff). ACM.

Pugh, S. D. (2001). Service with a smile: Emotional contagion in the service encounter. *Academy of management journal*, *44*(5), 1018-1027.

Radle, K., & Young, S. (2001). Partnering usability with development: How three organizations succeeded. *IEEE Software*, *18*(1), 38-45.

Rajanen, M., Iivari, N., & Anttila, K. (2011). Introducing Usability Activities into Open Source Software Development Projects–Searching for a Suitable Approach. *Journal of Information Technology Theory and Application*, *12*(4), 5-26.

Rapson, R. L., Hatfield, E., & Cacioppo, J. T. (1993). Emotional contagion.*Studies in emotion and social interaction, Cambridge University Press, Cambridge*.

Rasch, R. H., & Tosi, H. L. (1992). Factors Affecting Software Developers' Performance: An Integrated Approach. *MIS quarterly*, *16*(3).

Reel, J. S. (1999). Critical success factors in software projects. *Software, IEEE*, *16*(3), 18-23.

Rosenbaum, S., Rohn, J. A., & Humburg, J. (2000, April). A toolkit for strategic usability: results from workshops, panels, and surveys. In *Proceedings of the SIGCHI conference on Human factors in computing systems* (pp. 337-344). ACM.

Ross, S. M., & Morrison, G. R. (1996). Experimental research methods. Handbook of research for educational communications and technology: A project of the association for educational communications and technology, 1148-1170.

Rubin, J., & Chisnell, D. (2008). *Handbook of usability testing: how to plan, design, and conduct effective tests*. John Wiley & Sons.

Scheuren, F. (2004, June). What is a Survey?. American Statistical Association.

Schoenewolf, G. (1990). Emotional contagion: Behavioral induction in individuals and groups. *Modern Psychoanalysis*.

Seffah, A., & Metzker, E. (2004). The obstacles and myths of usability and software engineering. *Communications of the ACM*, *47*(12), 71-76.

Singer, T., & Lamm, C. (2009). The social neuroscience of empathy. *Annals of the New York Academy of Sciences*, *1156*(1), 81-96.

Sivaji, A., Abdullah, M. R., Downe, A. G., & Ahmad, W. F. W. (2013, April). Hybrid Usability Methodology: Integrating Heuristic Evaluation with Laboratory Testing across the Software Development Lifecycle. In *Information Technology: New Generations (ITNG), 2013 Tenth International Conference on* (pp. 375-383). IEEE.

Skov, M. B., & Stage, J. (2012). Training software developers and designers to conduct usability evaluations. *Behaviour & Information Technology*, *31*(4), 425-435.

Sohaib, O., & Khan, K. (2010, June). Integrating usability engineering and agile software development: a literature review. In *Computer design and applications (ICCDA), 2010 international conference on* (Vol. 2, pp. V2-32). IEEE.

Strauss, A., & Corbin, J. (1998). Basics of qualitative research: Procedures and techniques for developing grounded theory. *ed: Thousand Oaks, CA: Sage*.

Taft, D. K. (2007). Programming grads meet a skills gap in the real world.

Tullis, T., Fleischman, S., McNulty, M., Cianchette, C., & Bergel, M. (2002, July). An empirical comparison of lab and remote usability testing of web sites. In *Usability Professionals Association Conference*.

Usability Professionals Association (2012). Usability Body of Knowledge. Available from: http://www.usabilitybok.org/ [Accessed 3 January 2014]

Vidich, A. J., & Shapiro, G. (1955). A comparison of participant observation and survey data. *American Sociological Review*, 28-33.

Wehmeier, S. (2007). New oxford advanced learner's dictionary.

William R.. Shadish, Cook, T. D., & Campbell, D. T. (2002). *Experimental and quasi-experimental designs for generalized causal inference*. Wadsworth Cengage learning.

Wynekoop, J. L., & Conger, S. A. (1992). *A review of computer aided software engineering research methods*. Department of Statistics and Computer Information Systems, School of Business and Public Administration, Bernard M. Baruch College of the City University of New York.

Yin, R. (2003). K.(2003). Case study research: Design and methods. *Sage Publications, Inc*, *5*, 11.

Appendix A – Paper Contributions

This appendix contains the six paper contributions in their full versions. The papers have been published as follows:

1. Lizano, F., Sandoval, M. M., Bruun, A., & Stage, J. Usability Evaluation in a Digitally Emerging Country: A Survey Study. In Human-Computer Interaction–INTERACT 2013 (pp. 298-305). Springer Berlin Heidelberg (2013)
2. Lizano, F., Sandoval, M. M., Bruun, A., & Stage, J. Is Usability Evaluation Important: The Perspective of Novice Software Developers. In the British Computer Society Human Computer Interaction Conference (2013).
3. Lizano, F. & Stage, J. Improvement of novice software developers' understanding about usability: the role of empathy toward users as a case of emotional contagion. Accepted for publication in Proceedings of the 16th International Conference on Human-Computer Interaction (HCII) (2014)
4. Lizano, F., Sandoval, M. M., & Stage, J. Integrating Usability Evaluations into Scrum: A Case Study Based on Remote Synchronous User Testing. Accepted for publication in Proceedings of the 16th International Conference on Human-Computer Interaction (HCII) (2014)
5. Lizano, F. & Stage, J. I see you: Increasing Empathy toward Users' Needs. In Proceedings of the 37th Information Systems Research Seminar in Scandinavia (IRIS 2014).
6. Lizano, F. & Stage, J. Usability Evaluations for Everybody, Everywhere: A field study on Remote Synchronous Testing in Realistic Development Contexts. Accepted for publication in Proceedings of the 8th International Conference on Digital Society (ICDS) (2014)

Usability Evaluation in a Digitally Emerging Country: a survey study

Fulvio Lizano[1], Maria Marta Sandoval[2], Anders Brunn[1] and Jan Stage[1]

[1] Aalborg University, Department of Computer Science, Selma Lagerlöfs Vej 300, Aalborg East, Denmark

[2] National University, Informatics School, PO-Box 86-3000, Heredia, Costa Rica

{fulvio, bruun, jans}@cs.aau.dk, msandova@una.cr

ABSTRACT. Several emerging countries experience increasing software development activities. With the purpose of provide useful feedback on possible courses of action for increasing application of usability evaluation in such countries, this paper explores the status of usability evaluation in a digitally emerging country. Our aim is to identifying common characteristics or behavioral patterns that could be compared with digitally advanced countries. We used an online survey answered by 26 software development organizations, which gave a snapshot of the application of usability evaluation in these organizations. We found many similarities with advanced countries, several completely new obstacles more connected with software development matters and a relatively positive improvement in the lack of "usability culture". These findings suggest good conditions to improve conduction of usability evaluations in digitally emerging countries.

Keywords: Usability evaluation, advantages, obstacles, digitally emerging countries.

1 INTRODUCTION

Usability evaluation is a relevant and strategic activity in software projects [6]. For the user, a high level of usability in a software system is important [8]. For the user organization, usability is important because it can provide benefits such as increased sales, productivity, lower training costs and reduced technical support for users [2].

Previous studies of the perception of usability evaluation have been focused on obstacles and advantages. Two specific cases have identified obstacles in software organizations. The first one was a survey in Northern Jutland, Denmark [2] (known in this paper as the "D-Study") and the second, which replicated the first, was made in Southern Italy [1] (known in this paper as the "I-Study"). The D-Study identified several obstacles to increased application of usability evaluation, e.g. developer mindset, resource demands, lack of understanding, customer participation, conducting tests and test participants [2]. In the case of the I-Study, major obstacles identified were

resource demands, no suitable methods, developers' mindset and user availability. In addition, this study identified advantages of usability evaluation such as quality improvement, the users' satisfaction, resource saving and competitiveness [1].

These facts, together with other problems found in digitally advanced countries [3, 4, 5, 7, 9, 10, 11], can be visualized, as a whole, in a view formed by several dimensions of types of actors (e.g. users or clients, software developers, and organizations) plus types of facts (e.g. facts related to understanding, advantages and obstacles of usability evaluation) This is a way to put into context the real implications for usability evaluation in digitally advanced countries. In Table 1 we present the distribution of these findings into the cited dimensions (some references are related with different dimensions at once).

Table 1. Main findings related to usability evaluation.

	Types of facts (related to…)		
Types of actors	Understanding of usability evaluation	Advantages of usability evaluation	Obstacles of usability evaluation
Users / Customers	– User involvement[1,2] – Customer involvement[2]	– User satisfaction[1]	– Test participants[2] – Customer participation[2] – User availability[1]
Software developers			– Developer mindset[2] – Developer mindset[1] – Lack of trained eng.usab/HCI[10] – Personal developers tools[11]
Organizations	– Functionality[2] – Problem/task solving[2] – Possibility test[2] – Usability evaluation[1] – Usability definition[1] – Accessibility test[1]	– Qlty.improvement[1] – Competitiveness[1] – Resource saving[1]	– Lack of understanding[2,10, 11] – Resource demands[2] – Conducting tests[2] – No suitable methods[1] – Resource demands[1, 10] – Resistance to UCD-Usability[10] – Lack of comms. of impact[10] – Lack coupling UCD & S-Dev l.cycle[11] – Gap SD & usability[11] – Edu. lack coupling (SD&usab.) [11] – Lack of respect and support[5] – Limited description HCI in SE[4] – Strong differences HCI & SE[7] – Lack of "usability culture" [3, 9]

In this paper, we present the results of a study that explored the application of usability evaluation in software development organizations in a digitally emerging country. The aim of our study was to explore the understanding of the usability evaluation

concept, their advantages and obstacles. Our interest was identifying similarities, differences and patters that could enhance the application of usability evaluation in other digitally emerging countries. This explains why we will present our results and compare them to other studies, especially with the D-Study and the I-Study.

2 METHOD

2.1 Settings

We conducted a survey at software development companies in Costa Rica. According The Global Information Technology Report 2012 (World Economic Forum – www.weforum.org/gitr), Costa Rica is a digitally emerging country ranked in the 58 position of the Networked Readiness Index (NRI). Costa Rica has a NRI of 4 in a 1-to-7 scale.

2.2 Participants and procedure

The study has involved companies that could potentially conduct usability evaluations and in addition met the specific criteria, e.g. located in a specific geographical area (Costa Rica), that develop software or hardware with graphical user interfaces, that develop software for customers or for internal use and that employ more than a single person. Initial set of participants was made using the list of organizations affiliated to the Chamber of Information and Communication Technology - CAMTIC, by its Spanish acronym (www.camtic.org) (148 organizations). This organization is open to any IT organization of Costa Rica and was founded in 1998. Because CAMTIC is open to a broad range of IT companies, we decided to filter the original list obtaining a final list of 35 organizations. Our survey was completed by 26 organizations (74%). The average number of years of operation of the organizations participating in the study was 11. The average of age of the persons, who filled the questionnaire, was 39. 15% of them were females. All these companies were located at the central valley of Costa Rica (the most developed zone of Costa Rica). The organizations had this distribution on number of employees: 58% (1-10), 19% (11-50) and 23% (51-250). In order to find the person most appropriate to participate in the study, we contacted every company personally by phone in order to enquire who could provide an opinion that could reflect the position of the company in the survey. These persons received an electronic token to access the online survey which was active for 4 weeks.

2.3 Data collection and analysis

The questionnaire used in the study contained several parts. The main parts were: demographic and general information, products/services provided by the organization, methodology used to develop software applications, understanding of the usability evaluation concept, and obstacles and advantages of the application of usability evaluation. We used a combination of open-closed questions was used. The aim of open

questions was to permit participants to express themselves in their own words. Closed questions were used in order to allow to them to reconfirm data previously provided. We used different analytical approaches to analyze the data generated in the closed and open questions. A quantitative analysis was used on the closed questions, and grounded theory by Strauss and Corbin [12] was used for the analysis of the open questions.

3 RESULTS

We were interested in exploring the understanding of the concept of usability evaluation in the software development organizations. Our results allowed us to identify several categories of understanding that the organization have about this concept. In order to verifying what the organizations had understood by usability evaluation, after provide their definition of usability evaluation, we showed them a definition of usability evaluation based on the ISO-9241 standard. Next, we asked them if they made usability evaluations in their companies in accordance with this ISO definition and the strategy followed by them to do it. The participants basically reported two categories of strategy: internal or external conduction of usability evaluation. However, a relevant number of participants reported do not conduct usability evaluations at all. Finally, some participants did not provide a response for this enquiry. In Table 2, we present these results.

Table 2. Distribution of the strategy used to conduct usability evaluation related to the understanding of the concept of usability evaluation

	Category of understanding (In terms of…)				
Strategy used to conduct usability evaluation	Usability concept	Usability evaluation concept	Another kind of testing	No response	Percent
Internally	4	8	4		62%
Externally		1	2		12%
No Usab.evaluation	1	2	2		19%
No response				2	8%
Percent	19%	42%	31%	8%	

On the other hand, following the methodological approach established in our method, we identified the main advantages and obstacles for applying usability evaluation. First, using an open question we identified advantages and obstacles of the application of usability evaluation. Some participants offered more than one advantage/obstacle and others did not respond. The results were grouped in different categories of advantages/obstacles. In the case of the advantages, main results were product quality (35%), user acceptance (32%) and no advantages (32%). The main obstacles detected were users (22%), software design (19%), software development method (15%), costs (11%), software developers (4%) and no obstacles (30%). The organizations which reported do not conduct usability evaluation in their process, were the ones which had not found advantages or obstacles.

In order to complement the previous results, we presented to the participants with a list of common advantages and obstacles of usability evaluation. The participants could select more than one option. Results are showed in Table 3.

Table 3. Advantages and obstacles provided through closed questions.

Advantages.	#	P	Obstacles	#	P
User satisfaction	19	39%	Recruitment of test participants	9	20%
Quality improvement	18	37%	Conduct test / no method	6	13%
Competitiveness	5	10%	Developer mindset	17	38%
Resource saving	6	12%	Resources demands	8	18%
Other	1	2%	Other	5	11%
Total	49			45	

These results represent some of the main facts related to usability evaluation in digitally emerging countries. To facilitate a comparison of these facts with the ones presented in digitally advanced countries, we present them in a similar way as we did at table 1. Thus, as it is possible to see in Table 4, users, customers and software developers are not presented into the understanding of usability evaluation, by the software development organizations at the digitally emerging countries. In addition, our results suggest a weak visualization of advantages and obstacles in the same context. Similar to the digitally advanced countries, at digitally emerging countries it is possible to find more facts related to the understanding, advantages and obstacles of usability evaluation, in the context of the organizations.

Table 4. Summary of results.

	Types de facts (related to…)		
Types of actors	Understanding of usability evaluation	Advantages of usability evaluation	Obstacles of usability evaluation
Users / Customers		– User acceptance – User satisfaction	– User – Test participants
Soft.Dev.			– Software developers mindset
Organizations	– Usability concept – Usability evaluation concept – Other test	– Product quality – Qlty improvement – Competitiveness – Resource saving	– Software design – Software development method – Costs – Usability evaluation conduction. – No usability evaluation method

4 DISCUSSION

Our results suggest a relatively good understanding of the understanding of the usability evaluation concept, including some similarities to previous studies, e.g. in some

aspects of the notion of usability [1], and in usability matters as a whole, specifically in some responses related to user involvement [2]. The good understanding about the definition of usability evaluation was obtained from organizations which conduct usability evaluation internally. This practical experience supported this better understanding. This is even more evident when we analyze the reasons given by those organizations that do not conduct usability evaluations (19%). Although the distribution of their understanding is uniform in the different categories of understanding, these participants provided opinions that are clear signals of a misunderstanding about usability evaluation, e.g. "in open source software projects you do not need usability evaluations" or "some projects do not require usability evaluations" and "a software project only needs functional tests". Here, we can see and excellent example of what the lack of "usability culture" is [3, 9].

In the case of advantages, our results are fully in agreement with the I-Study [1]. However, in the case of the obstacles, our study found very interesting results. The 'user' was identified as one of the most relevant obstacle. This was emphasized by participants who conduct usability evaluations internally, which makes this result conclusive. Both the D-Study and I-Study also identified this obstacle but with a lower level of importance [1, 2]. This finding allows us to notice that in a digitally emerging country, participation of users in usability evaluation seems to be particularly challenging. Consistently with the D-Study and the I-Study, our study confirmed obstacles related to resource demands and software developers' mindset [1, 2]. It is interesting to notice that the level of relevance given to this last obstacle has changed across the D-Study, the I-Study and our study. This obstacle was very important in the D-Study. In the I-Study, its relevance was lower. Finally, in our study this obstacle is the last one mentioned by the participants. This change could, initially, reinforce our perception of a positive change in the lack of a "usability culture". However, other results obtained in the closed questions seem to offers contradictory results. In this case, the most important obstacle selected by participants was related to the software developers' mindset problem. These different levels of relevance are not necessarily a contradiction. Actually, the fact that this obstacle was cited twice in our study allow us to conclude that this matter continues been one of the most recognized obstacles against increased use of usability evaluation. The second obstacle identified in this part of our study is related to resource demands, which is not surprising.

In addition, there are new obstacles that were identified in our study. First obstacle is related to problems in the design process of the software, which subsequently could hinder conduction of usability evaluations. Second, a new obstacle was identified in some problems related to the software development method. This obstacle was identified by the I-Study in 2011 but not by the D-Study in 2008. Here it is possible to observe a change of tendency in the lack of "usability culture", into those organizations that have practical experience; an alternative view about the new obstacles, which is more connected to the software development process, seems to emerge to reduce some problems such as the confusion, the lack of coupling and some gaps between SE and HCI [11].

In digitally advanced countries the main facts related to understanding, advantages and obstacles to conduct usability evaluation are more connected with methodology

and the organization (see Table 1). Users, customers and software developers have a lower visualization. More remarkable is the fact that software developers are not presented at all in such dimensions. Only in the case of the obstacles, it is possible to find more facts related to users, customers and developers.

In digitally emerging countries, this situation seems no to be better (see Table 4). Into the understanding of usability evaluation, the users, the clients and the developers are excluded at all. Only a limited number of advantages were noticed for users and clients, none related to developers.

We think that our study provide interesting results that can be extended to other similar contexts. The digitalization level and other human and economical indicators are pretty similar to others countries in the same region, e.g. Ecuador, Trinidad & Tobago, Panama, Peru, Brazil and The Bolivarian Republic of Venezuela. The average on Networked Readiness Index (NRI) in these countries (including Costa Rica) is 3,71 (SD=0,32), the mean value for the GNI per capita in PPP terms (constant 2005 international $) is 12,051 (SD= 4,740), the average of expected years of schooling is 9,05 (SD=2,21), the mean value for expectancy of life (years) is 74,81 (SD= 2,79). Main differences are related to population and territorial extension. Considering all these facts, the context studied in our research can be considered a good referent about how usability evaluation is conducted in other digitally emerging countries.

5 Conclusion and future works

In this research we have explored the application of usability evaluation in software development organizations in the digitally emerging countries. To accomplish this, we conducted a questionnaire survey with 26 participating software development organizations. As part of our research, our findings were contrasted with results from similar studies in digitally advanced countries. The aim of our study was to obtain valuable feedback that could orientate future enhancement actions of application of usability evaluation in digitally emerging countries.

Our study found a relatively acceptable conduction of usability evaluation in digitally emerging countries, embodied by a fairly clear understanding about the meaning of usability evaluation and similar advantages and obstacles to the found in other digitally advanced countries. In addition, our research has identified new obstacles such as the users' behavior and problems related to the design of the software. These new obstacles can offer to HCI theory a complementary perspective on usability evaluation. These new findings seem to imply a decreasing tendency in the lack of "usability culture". However, our results do not permit strong conclusions about this matter as it was not a focus of our study.

However, any improvement of conduction of usability evaluation at the context studied must necessarily go through an empowerment process of users, clients and software developers, as main actors in such processes. In the case of users and clients, reasons to do that are more than evident; in some sense, these actors are a main cornerstone of theory of HCI. For developers, this strategy should help continuing improvement of some well studied problems, e.g. confusion, the lack of coupling and the

gaps between software engineering and HCI. Future works could focus on exploring specific forms to enhance and increase the use of usability evaluations in software development organizations located in digitally emerging countries.

Acknowledgments

The research behind this paper was partly financed by Universidad Nacional (Costa Rica), MICIT (Costa Rica), CONICIT (Costa Rica), and the Danish Research Councils (grant number 09-065143).

References

1. Ardito, C., Buono, P., Caivano, D., Costabile, M.F., Lanzilotti, R., Bruun, A., Stage, J.: Usability Evaluation: a survey of software development organizations. In Proceedings of 33 International Conference on Software Engineering & Knowledge Engineering. Miami, FL, USA (2011)
2. Bak, J.O., Nguten, K., Risgaard, P., Stage, J.: Obstacles to Usability Evaluation in Practice: A Survey of Software Development Organizations. In Proceedings of the 5th Nordic conference on Human-computer interaction: building bridges, pp.23-32. ACM, New York, NY, USA (2008)
3. BugHuntress QA Lab: Mobile usability testing Problems and solutions. Research Report, Quality Assurance: Management & Technologies (QAMT), Ukraine. (2007)
4. Ferre, X., Juristo, N., Moreno, A.M.: Obstacles for the Integration of HCI Practices into Software Engineering Development Processes. C. Ghaoui & Idea Group Reference (Eds.). Encyclopedia of Human Computer Interaction, pp. 422-42 (2006).
5. Gulliksen, J., Boivie, I., Persson, J., Hektor, A., Herulf, L.: Making a difference: a survey of the usability profession in Sweden. In Proceedings of the third Nordic conference on Human-computer interaction (NordiCHI '04), pp 207-215, ACM, New York, USA (2004).
6. IEEE Computer Society: SWEBOK Guide to the Software Engineering Body of Knowledge, http://www.computer.org/portal/web/swebok (2004)
7. Juristo, N., Ferre, X.: How to integrate usability into the software development process. In Proceedings of the 28th international conference on Software engineering (ICSE '06), pp 1079-1080, ACM, New York, NY, USA (2006)
8. Lindgaard, G., Chattratichart, J.: Usability Testing: What Have We Overlooked?. In CHI 2007 Proceedings, Usability Evaluation. San Jose, CA, USA. (2007)
9. Muzaffar, A., Azam, F., Anwar, H., Khan, A.: Usability Aspects in Pervasive Computing: Needs and Challenges. International Journal of Computer Applications, 32(10):18-24, October 2011. Published by Foundation of Computer Science, New York, USA. (2011)
10. Rosenbaum, S., Rohn, J.A., Humburg, J.: A toolkit for strategic usability: results from workshops, panels, and surveys. In Proceedings of the SIGCHI conference on Human factors in computing systems (CHI '00), pp 337-344, ACM, New York,USA, (2000)
11. Seffah, A., Metzker, E.: The obstacles and myths of usability and software engineering. Commun. ACM, 47, pp 71-76, December (2004).
12. Strauss, A., Corbin, J.: Basics of qualitative research. Techniques and Procedures for Developing Grounded Theory, 2.edition. SAGE Publications, (1998)

Is Usability Evaluation Important: The Perspective of Novice Software Developers

Fulvio Lizano
Aalborg University
Dept. of CS
Selma Lagerlöfs Vej 300
Aalborg East, Denmark
fulvio@cs.aau.dk

Maria M. Sandoval
National University
Informatics School
PO-Box 86-3000
Heredia, Costa Rica
msandova@una.cr

Anders Bruun
Aalborg University
Dept. of CS
Selma Lagerlöfs Vej 300
Aalborg East, Denmark
bruun@cs.aau.dk

Jan Stage
Aalborg University
Dept. of CS
Selma Lagerlöfs Vej 300
Aalborg East, Denmark
jans@cs.aau.dk

In this paper we present the results of a study which aims to explore the perspective of novice software developers about usability evaluation. It is important for a software organization to understand how novice developers perceive the role and importance of usability evaluation. This will permit development of effective methods and training programs that could potentially increase the application of usability evaluation. The results suggest that the perspectives of novice software developers about usability are characterized by a clear understanding about what usability evaluation is and a clear awareness about obstacles and advantages. However, our study also reveals certain shortcomings in the "usability culture" of novice developers, especially about the users' role in usability evaluation. Despite this limited "usability culture", novice developers' understanding of usability evaluation reflects a positive opinion about their participation in these activities. In addition, novice developers think that usability, in a general sense, is an important aspect of their work.

Usability evaluation, usability evaluation perspectives, novice software developers

1. INTRODUCTION

Usability evaluation is an important and strategic activity in software projects (IEEE Computer Society, 2004). Its relevance had been recognized in the context of the user (Lindgaard & Chattratichart, 2007) and the software organization (Bak et al., 2008). However, several studies had identified important obstacles to its applicacion in software developement process (Bak et al., 2008; Ardito et al., 2011). Some of these obstacles are related to the understanding of the usability concept, resource demands, the lack of suitable methods, availability of users and the software developers' mind-set (e.g. it is difficult to think like a user, lower acceptance of usability evaluations, and developers' emphasis in implementing efficient code)..

Alternatively, Rosenbaum and Rohn & Humburg (2000) reported other obstacles such as resource constraints, resistance to "User-Centered Design/usability", lack of understanding/knowledge about the usability concept, lack of better ways to communicate the impact of work and results, and lack of trained engineers in usability/HCI. In a similar way, Seffah & Metzker (2004) identified problems such as misunderstanding the concept of usability, lack of coupling between User-Centered Design techniques and software development life cycle, the gap between software development and usability, and the fact that education about software development is not coupled with usability. In addition, Gulliksen et al. (2004) argued that the main obstacle is the lack of respect and support for usability issues and its practitioners. Finally, Ferre & Juristo & Moreno (2006) argue that a diffuse positioning of HCI techniques in the software development process is the main obstacle presented to usability.

All of these studies have considered software developers as one homogeneous group. However, there are obviously clear differences between novice and expert software developers. Usually, an expert developer has several years of experience not only in technical activities as for instance coding, but also in other roles, e.g. architect, project manager, etc. (Berlin, 1993; Roff & Roff, 2001). The professional growth process of novice developers is characterized by a continuous learning process both in their formal education at college and their new professional roles in organizations. However, in their academic process it is remarkable the absence of training for soft skills which are a major component in the new jobs. (Begel & Simon, 2008). This fact could explain why according Taft (2007) there are some particular

problems of these new college graduates such as the lack of communication and team work skills, as well as limited experience in complex development processes, legacy code, deadlines, and working with limited resources.

The literature presented above conveys a good understanding of specific soft skills of novice software developers. Yet none of the studies have dealt with novice software developers' perception of usability. This information is crucial in order to develop adequate methods, which enable effective participation of novice developers and facilitate their interaction with more experienced developers. Such knowledge could also help in the design of adequate training programs for novice developers.

This paper presents the results of a study that explored the perspective about usability evaluation of novice software developers. We studied the understanding of the concept of usability evaluation and the obstacles and advantages for apply usability evaluation as they were seen by novice software developers. To complement this particular perspective, the study also explored the importance given by novice developers to usability in a general sense. This paper presents the method used, the results, an analysis section, and finally our conclusion.

2. METHOD

Our study used an online questionnaire with participation of advanced students of a System Engineering undergraduate course.

2.1 Participants

We focussed on advanced students enrolled in the last core course of System Engineering. These students have 18 months of real experience working in a software project with real users. In addition, because of particular characteristics presented in the context where the study was made, 87% of these students normally have a job related with software development processes (Lizano & Sandoval & García, 2008). Finally, the lack of training for soft skills presented in academic organizations (Begel & Simon, 2008), equally affects both advanced students and novice software developers. Combination of previous courses and modest real professional experience has produced in these participants a particular perspective that we were interested in explore.

We contacted the participants through the official list of students and projects. The questionnaire was submitted to 141 students included in the official register of the course. 72 completed it (51%). The average age is 22.2 (SD =2.17). 21 females (29%) participated in the study. All participants lived and worked in the Central Valley, which is the most developed zone in Costa Rica. The organizations where the participants had their jobs or where they carried out their project, had the following sizes: 26% (1-10 employees), 26% (11-50 employees), 17% (51-250 employees) and 31% (>250 employees).

2.2 Procedure

We contacted all the professors who lectured on the last core course of System Engineering in order to explain the motivation behind the study and request their collaboration. All professors then relayed the information to the students. Each student received instructions on filling in the questionnaire with focus on their role as software developers in an organization or as members of a software team that developed a software system in an organization during the previous18 months.

2.3 Data collection and analysis

The questionnaire was divided into sections such as demographic and general information, importance of usability, understanding of the usability concept, obstacles and advantages of usability evaluation.

The importance of usability issues given by participants was measured using questions grouped in five concept pair. Each pair was formed by two topics, one of them related to software development activities, e.g. "identify potential software problems and bugs" and the other related to usability activities, e.g. "identify potential usability problems". For each pair of concepts, the participants had to select which topic was more important. The concepts were defined based on the main contents of a course in systems engineering (software development topics) and of a course in design, implementation and evaluation of user interfaces (usability topics). The order of the pair of concepts and the position of each concept into the pair, were randomly defined. Two-alternative forced choice was used in order to contrast usability and software development matters. In this sense, our aim was focussed on the relation of usability and common software development matters in the context of novice software developers, which is the logical alternative considering limitation of experience of such developers.

Data on obstacles to usability evaluation were collected by using a combination of open/closed questions. First, an open question was used to allow participants to express an obstacle, using their own words. These open questions allowed us triangulate the results obtained in the closed questions cited above and, in some way, reduce bias of such ipsative questions by offering opportunity to participants to clarify or express in a different way their opinions. Next, a closed

question with several options of commonly known obstacles was presented. The idea was to offer alternative obstacles that the participants had not considered before. The common obstacles were defined based on Bak et al. (2008) and Ardito et al. (2011). We used a similar approach to collect data about advantages of usability evaluations.

We used two different approaches to analyse the data collected. A quantitative analysis was used for the closed questions, while we used the grounded theory approach for the open questions (Strauss & Corbin, 1998).

3. RESULTS

3.1 The importance of usability

We wanted to know how the novice software developers perceived the importance of usability in a broad sense. We presented to the participants several pairs of concepts in order to inquire which one they found most important. The results are presented in Table 1.

Table 1: *Perceptions of the importance of usability versus software development activities*

#	Detail	# Res	%	Dif.
1	*Usability. of soft (U)* *Dev.Quality code (S)*	*41* *31*	*57%* *43%*	*↑14*
2	*Des.bas. U. needs(U)* *Des.bas.requer. (S)*	*47* *25*	*65%* *35%*	*↑30*
3	*Identify usab. prob.(U)* *Identify bugs (S)*	*26* *46*	*36%* *64%*	*↑28*
4	*HCI (U)* *SQA (S)*	*10* *62*	*14%* *86%*	*↑72*
5	*Des.consid. VDP (U)* *Des.consid. patterns(S)*	*15* *57*	*21%* *79%*	*↑58*
AVG	*Usability concepts (U)* *Soft.Dev. concepts (S)*	*28* *44*	*39%* *61%*	*↑22*

U: Concept/activity related with usability
S: Concept/activity related with other software process

Despite preference on usability in the first two pairs of concepts it is evident that for the novice software developers, technical quality is the primary goal in software development. Overall, the novice software developers find software development activities more important than usability activities (the overall average of perceived importance was 61% versus 39%). The differences are largest in the pairs where usability is contrasted with some software activity related to quality, i.e. the largest difference is in pair 4 where the usability topic presented was HCI. This fact could be originated by certain unawareness about HCI, but it seems as it is mostly related to the preference that novice software developers have for software matters especially by software quality.

3.2 Understanding of the usability evaluation concept

An additional aim of this research was to explore the understanding of the usability evaluation concept among novice software developers. We decided to use an open question in order to obtain these data. 44 of the 72 respondents expressed their understanding in a way clearly related with a generally accepted definition of usability evaluation. In their answers it is possible to find references to concepts such as “user”, “test” and “usability”. For example, an answer that could illustrate this understanding is: *“It is tests that measure how well a user can use a program, without requiring any external intervention”*.

Fewer participants (11 of 72) responded using concepts more related to functionality, e.g. *“It is the tests made with the end user to verify the functionality of the software, to find and fix errors”* Some other participants (8 of 72) expressed understandings related to other types of testing.

After this open question, we presented to the participants with a definition of usability evaluation based on the ISO-9241 standard. The idea was to explore if the novice developers really found that they had participated in or worked with usability evaluation, according to that particular definition. In general, most novice developers found that they had participated in conducting a usability evaluation; 40 of 72 participants (56%) expressed a high level of agreement on that. Only 2 of 72 participants (2.8%) expressed a high level of disagreement. This result corresponds to the clear understanding that participants have about the usability evaluation concept. In their definitions about what usability evaluation is, the novice developers present concepts or ideas that know by first hand due their participation in these kinds of evaluations.

3.3 Obstacles in conducting usability evaluations

We used a combination of open/closed questions to identify perceived obstacles to the application of usability evaluation according to the novice developers. First, an open question was presented inquiring about obstacles or problems that the respondents had experienced during a usability evaluation. They were requested to write down one or more obstacles or problems. One participant mentioned 2 obstacles, while the rest only mentioned one. Thus the total number of obstacles or problems mentioned was 73. The primary obstacle detected is related with users’ behaviour/problems. We identified this obstacle in 23 of 73 items. Next example illustrate this result: “The software is not accepted by the user, even considering that this software is what he had

requested". As it is possible to see in this example, the users' behaviour is presented in a negative context in the software development process. In the second place, we found two different obstacles not necessarily related to usability evaluations. We identified both obstacles in 13 of 73 items. In the first case, participants mentioned problems in software that are directly or closely related to its design (e.g. "There are design factors that the user does not like, or technical details that the user wants in the system"). The second case is related to technical and organizational issues (for example "Problems in the software (bugs), problems with the data (e.g. clean databases)")

In the closed question we presented to participants several obstacles previously identified in the literature. Here the most selected obstacle was "too many resources" (28 of 122). This result justify our intention to offer other options of common obstacles that might have gone unnoticed in the open question; this obstacle was not mentioned in the open question, but in the closed section it was the most selected option. The second most selected obstacle was the difficulty to get customers/users to participate in usability evaluations (23 of 122). These results are closely related to the first obstacle detected in the open question (users' behaviour or problems). In third position we found two obstacles: my software does not have any problems (17 of 122) and no usability problems (16 of 122). These obstacles, which are connected each other, show that for a considerable number of the novice developers, their software does not have problems.

3.4 Advantages in conducting usability evaluations

We applied same combination of open/closed questions to identify perceived advantages to the application of usability evaluation according to the novice developers. The total number of advantages mentioned was 79. The primary advantage mentioned by the respondents was an increase of quality in the software (26 of 79) for example "Allows for fixing problems that could become more serious if they are not repaired on time". In this case, it is possible to reconfirm the participants' perspective about the relevance of quality (see section 3.1). The second most mentioned advantage is a guarantee benefit of usability evaluations: it improves the software development method. 24 of 79 responses were related to this advantage, e.g. "Creation of a system that will be controlled by and adapted to the enterprise processes in an easy way". According these comments, novice software developers seem at usability evaluations as a way of to identify any potential usability problems and incidentally, improve other relevant aspects of the software development process. Other advantages cited by participants were users' satisfaction (16 of 79), improve the design of the software system (6 of 79), and professional growth (5 of 79).

The closed question, which contained several options of commonly accepted advantages, generated results closely related to the previous ones. The two primary advantages were "increase user satisfaction" and "quality improvement". They were widely selected (68 and 61 respectively of 211). The third most selected advantage was "increase competitiveness" which was selected by 33 participants. Another relevant advantage is to increase competences, which was selected by 31 respondents. In this case, novice software developers think that their participation in a usability evaluation could help them increase their professional competences. This is another new finding of this study.

4. DISCUSSION

The aim of this study was to explore the perspectives of novice software developers on usability evaluation. We focussed on the perceived importance of usability for novice developers, on their understanding of the usability evaluation concept and on obstacles to and advantages of conducting usability evaluations.

Concerning the importance of usability, our study showed that 39% of the novice developers perceive usability topics as being more important than software development topics. Given the situation with a lack of "usability culture" and the perceived obstacles to applying usability evaluations, it is interesting that more than one third of the novice developers still find usability most important. This indicates that usability has an impact on the mind-set of some of the novice software developers. In comparison, software development topics are considered more important by 61% of the respondents. Our results shows that sometimes usability activities are perceived as being more important than software development activities (see Table 1, pairs 1 and 2). By contrast, software development is more relevant than usability when contrasting usability activities against other quality activities (see Table 1, pairs 3, 4 and 5). In general, usability activities received relatively much emphasis; however, software quality is still the main focus for novice developers. This is particularly clear in concept pairs 4. Here, 86% of the participants considered software quality assurance as being more relevant than human-computer interaction. Thus we conclude that although usability is perceived by novice developers as being important, quality in software is even more so.

Our findings also show that novice developers can define usability evaluation quite well, i.e. they

understand the concept of usability evaluation. This is clear from the considerable number of participants who provided definitions of usability evaluation using concepts such as evaluation, user and usability in their answers. Moreover, only a few respondents used concepts related to functionality or various kinds of technical or functional tests. This proper understanding of the concept of usability evaluation can explain why novice developers are highly convinced of the relevance of their participation in this kind of evaluation. The clear understanding of the concept of usability evaluation contradicts results found in other studies (Ardito et al., 2011; Bak et al., 2008; Seffah & Metzker, 2004; Rosenbaum & Rohn & Humburg, 2000). Even if we consider that those previous studies had been made with more experienced actors (mainly from software organizations), the novice developers' clarity on these concepts originate from the education programs they have followed. Nowadays, HCI topics are common in many software development curricula. Contrastingly, we found a low level of understanding of the HCI concept, which has also been reported in other studies (Rosenbaum & Rohn & Humburg, 2000; Ferre & Juristo & Moreno, 2006)

Our findings regarding the perceived obstacles to applying usability evaluation show that the main obstacle is the users' behaviour and other problems related with users. Confirming this, the novice developers consider that their software does not have usability problems. These results indicate a lack of "usability culture". On the surface, this contrasts our findings of a high level of understanding of the usability evaluation concept; but as noted by Rosenbaum & Rohn & Humburg (2000), a clear understanding of the usability concept is not enough to understand what usability evaluation implies. Yet Nielsen (2005) reports contradicting findings by showing that users are strongly engaged in the usability testing process. Our findings indicates a manifestation of a well-known obstacle which is the software developers' mind-set.(Ardito et al., 2011; Bak et al., 2008). Another obstacle identified by novice developers relates to the perceived high cost of usability evaluations, which is also found in other studies (Ardito et al., 2011; Bak et al., 2008; Rosenbaum & Rohn & Humburg, 2000).

In addition, some new obstacles are suggested in our study. This is particularly the case with design of the software and other technical and organizational problems, e.g. software bugs, lack in "usability culture", etc. In some way, this new finding contradicts the decoupling of usability from software engineering reported by Seffah & Metzker (2004); for novice developers there are an evident relation between usability and other software development activities. However, the concern of novice developers for software bugs, illustrates the importance they contribute to software quality at the cost of usability issues (See Table 1)

The ability of usability activities to help improving software quality is considered to be the main advantage. This result confirms the importance of software quality for novice developers. This is an expected result considering the general opinions related to the aims of the testing process; it is generally accepted that testing is performed, among other major aims, to evaluating product quality (IEEE Computer Society, 2004). In addition, improved user satisfaction is another advantage identified by novice developers. These advantages are supported in the study of Ardito et al. (2011). This particular opinion of novice developers, related to one of the most relevant advantages of usability evaluation, seems to contradict their own perspective about the main obstacle: the user. This also indicates the lack of "usability culture" among novice developers.

Extending the findings of Ardito et al. (2011) about advantages of usability evaluation, our study identifies two new advantages: it could improve the software development method and developers' participation in usability evaluations could allow them to increase their professional competences. Certainly, usability evaluation has a clear purpose in identifying usability problems that would otherwise affect the software usability negatively. This could be considered the major aim of usability evaluations. However, it is interesting that the novice developers emphasize other benefits related to software development. With this, the novice developers present themselves as persons who try to see beyond obvious and expected results of a particular process as, in this case, usability evaluations. The increased competencies of novice developers allow us to understand that these professionals have criteria to recognise the knowledge is important for their future careers.

5. CONCLUSION

This paper presented the perspective of novice software developers on usability evaluation. This included several elements such as the importance, meaning, obstacles and advantages of usability evaluations. We have contrasted our study with other previous studies, which emphasize organizational perspectives.

Our study showed that usability activities are considered important by more than one third of the novice developers. Compared to usability activities, software quality activities have a higher priority. Despite this, usability still appears to be important. Emphasis on usability activities could be increased, e.g. with training programs that diminish the lack of "usability culture" detected in this study. In contrast

to the lack of "usability culture", our results also show that novice developers have a clear understanding of what usability evaluation is as well as they an advanced ability to express obstacles and advantages. Our findings about obstacles and advantages are supported by other studies. In addition, we have also found new ones. The role of design in usability evaluation is noteworthy, something that is relevant for novice developers as an obstacle and also as an advantage.

In general, the novice software developers' perspective could be contradictory with the belief of their emphasis in implementing efficient code. Our conclusion is that both approaches usability and efficient codification seems to be relevant for novice developers such is showed in their vision about role of usability and software quality.

For future work we would like to study the potential synergies between usability evaluation and design activities in order to help in the coupling efforts of software engineering and HCI.

ACKNOWLEDGEMENTS

The research behind this paper was partly financed by National University, MICIT, CONICIT (Costa Rica), and the Danish Research Councils (grant number 09-065143).

References

Ardito, C., Buono, P., Caivano, D., Costabile, M.F., Lanzilotti, R., Bruun A. and Stage, J. (2011). Usability Evaluation: a survey of software development organizations. In *Proceedings of 33 International Conference on Software Engineering & Knowledge Engineering*. Miami, FL, USA, 7-9 July 2011.

Bak, J.O., Nguten, K., Risgaard, P. and Stage, J. (2008). Obstacles to Usability Evaluation in Practice: A Survey of Software Development Organizations. In *Proceedings of the 5th Nordic conference on Human-computer interaction: building bridges*, Lund, Sweden, 20-22 October 2008. ACM, New York, NY, USA, 23-32.

Begel, A., and Simon, B.(2008). Novice software developers, all over again. In *Proceedings of the Fourth international Workshop on Computing Education Research (ICER '08),* Sydney, Australia, 6-7 September 2008. ACM, New York, NY, USA, 3-14.

Berlin, L. M. (1993). Beyond program understanding: A look at programming expertise in industry. In *Empirical Studies of Programmers: Fifth Workshop*. Ablex Publishing Corp., 6–25.

Ferre, X., Juristo, N. and Moreno, A.M. (2006). *Obstacles for the Integration of HCI Practices into Software Engineering Development Processes*. In C. Ghaoui & Idea Group Reference (Eds.). Encyclopedia of Human Computer Interaction, 422-42.

Gulliksen, J., & Boivie, I., & Persson, J., & Hektor, A. and & Herulf, L.(2004). Making a difference: a survey of the usability profession in Sweden. In *Proceedings of the third Nordic conference on Human-computer interaction (NordiCHI '04*). Tampere, Finland, 23-27 October 2004. ACM, New York, NY, USA, 207-215.

IEEE Computer Society (2004). *SWEBOK Guide to the Software Engineering Body of Knowledge*. Available from: www.swebok.org [01 May 2013]

Lindgaard, G., and Chattratichart, J. (2007). Usability Testing: What Have We Overlooked?. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. San Jose, CA, USA, April 28-May 3 2007. ACM, 2007. p. 1415-1424.

Lizano, F., Sandoval, M.M. and Garcia, M.A. (2008). Generación de una cultura organizacional en el aula: "un caso específico: la cultura hacia la calidad en el desarrollo del software, en la cátedra de Ingeniería de Sistemas. En *I Congreso Intern.Computación y Matemática.* UNA, Heredia, Costa Rica.

Nielsen, J., (2005) *Authentic Behavior in User Testing*. Jakob Nielsen's Alertbox. Available from: www.useit.com/alertbox/20050214.html [01-05-2013].

Roff, J. and Roff, K (2001). *Careers in E-Commerce Software Development.* The Rosen Publishing Group.

Rosenbaum, S., & Rohn, J.A., & Humburg, J. (2000). A toolkit for strategic usability: results from workshops, panels, and surveys. In *Proceedings of the SIGCHI conference on Human factors in computing systems (CHI '00),* The Hague, The Netherlands, April 1-6, 2000. ACM, New York. 337-344.

Seffah, A. and Metzker, E. (2004). The obstacles and myths of usability and software engineering. *Communications of the ACM*, 47(12), 71-76.

Strauss, A. and Corbin, J. (1998). *Basics of qualitative research. Techniques and Procedures for Developing Grounded Theory*, 2.edition. SAGE Publications.

Taft, D. K.(2007). Programming Grads Meet a Skills Gap in the Real World. *eWeek.com*. Available from: www.eweek.com/c/a/Application-Development/Programming-Grads-Meet-a-Skills-Gap-in-the-Real-World/ [01 May 2013].

Improvement of novice software developers' understanding about usability: the role of empathy toward users as a case of emotional contagion

Fulvio Lizano, Jan Stage

Aalborg University, Department of Computer Science, Aalborg, Denmark

{fulvio, jans}@cs.aau.dk

Abstract. There are several obstacles when it comes to integrating Human-Computer Interaction (HCI) activities into software development projects. In particular, a lack of understanding on the part of novice software developers regarding usability is one of the most cited problems related to this integration. Observation of usability evaluation by these developers has been cited in the literature as an alternative to improve their understanding about usability due to the fact that, among other things, this improves the level of empathy with users. In this paper we present the results of a quasi-experiment which explores the origin of this improvement. Our study suggests that the empathy of novice developers towards users could be originated by Emotional Contagion (EC) of these developers. This EC occurs unconsciously in activities where these developers can observe users working with the software. The present research is an initial approximation as to the relation which EC and empathy have in order to improve the novice software developers' understanding of usability.

Keywords: Software development · usability · understanding of usability · empathy towards users · emotional contagion.

1 INTRODUCTION

The lack of understanding on the part of novice software developer regarding usability, is one of the most cited problems about to integration of HCI activities (specially usability evaluations), into software development projects. [2, 3], [15], [18]. This problem suggests a low priority of software developers on the user. Developers' motivators confirm their focus on personal matters [9], [14].

According some studies, observation of usability evaluations by developers improves their understanding of usability and also their empathy with users [10], [21]. Other researchers confirm this increasing of empathy in contexts with close interaction with users [6, 7], [12, 13]. Causes of such phenomenon in developers have not been studied yet.

The empathy [5], [20] has its origin in an Emotional Contagion (EC) process [8], [17]. This process occurs between two actors: the observer and the observed. In the

pro-cess, the observer unconsciously acquires the emotions of the observed after seeing and interacting with him for some time [4], [20]. The observer assumes a submissive role in her/his interaction with the observed who, in turn, assumes a dominant role. The particular circumstances or personalities of each are decisive in establishing who assumes a particular role [17]. The EC-Process could be fundamental to explain why developers experiment an increasing in the empathy with users and also in the under-standing of usability, during their observation of usability evaluations.

Considering this, we conducted a quasi-experiment [23] which aimed to explore the improvement of the understanding of usability and also the empathy with users by novice software developers, into a usability evaluation context. Our study attempts to fill the gap in the literature by explaining this situation since a perspective of an EC-Process.

In the first section of this paper we present the introduction and a brief literature review. Next, the method is presented in the section 2. Following this, we present the results of our study. After the results have been summarized, the paper presents the discussion section before concluding with suggestions for future work.

2 METHOD

We conducted a quasi-experiment [19], [23] where nine developers (SE/CS students), grouped in two teams, conducted a usability evaluation with users [16]. Usability evaluations were used to set an interactive environment with users; our focus was on the improvement process of the understanding of usability, more than in the results of the tests.

We collected data related to the students' understanding of usability two weeks before the test (1DC) and immediately after (2DC) the test. Additionally, we held interviews with students. The aim of these interviews was to allow the authors to elaborate on or clarify some findings of the study.

In every DC we used two forms. The first form (F1) was used in order to allow the students to express their opinions related to the main strengths and weaknesses presented in their software. The second form (F2) was used to measure the relative importance given by the students to certain software/usability concepts. In this form, we used 5 pairs of concepts or sentences which could illustrate normal activities for SE or HCI practitioners.

The concepts related to SE were:

- Modelling software requirements.
- Understanding how a system is designed.
- Realizing how the Unified Modelling Language (UML) could be applied to a software project.
- Knowing about software modelling patterns.
- Understanding the main concepts of Object-Oriented modelling.

The concepts related to HCI were:

- Designing an interface both physically and conceptually correct.
- Understanding how a user interface could be designed.
- Realizing how the Gestalt Laws could be applied to a software project
- Knowing about visual design principles.
- Understanding the main concepts of Human-Computer Interaction.

The analysis of the data collected was focused on the identifying the improvement in the understanding of usability by analyzing differences (between 1DC and 2DC) in F1. In addition, we identified the understanding pattern of usability based on [1]. Results were triangulated with F2 and the interviews. As part of the analysis, we identified the origin of such improvement and the implications for the empathy toward users.

3 RESULTS

In this section we present the results of the study. We felt that in order to better understand the mechanism(s) of generation of empathy towards users, we first needed to establish in a general and detailed way, beyond doubt, a real improvement in the understanding of usability. Following this, we could identify and understand better the patterns which characterized this improvement. This explains why we first focused on describing the variations in the understanding of usability after applying the corrective action (conduction of usability evaluation by the students); these results are presented in Sections 3.1, 3.2 and 3.3. Next, in Section 3.4 we will present the patterns which characterized the improvement of the novice software developers' understanding of usability. In this part of the results, we also included some of the students' personal opinions given during the interviews, in order to complete the picture.

3.1 Overall understanding of usability.

We were interested in gauging the perceptions of students before and after their participation in the usability evaluation Table 1 presents the general results obtained when we enquired about the strengths (S) and the weaknesses (W) of their software (form F1).

Table 1. Strengths (S) and weaknesses (W) related to usability before and after conducting usability evaluation.

Facts	F1				Variance	
	1 DC		2 DC			
	S	W	S	W	S	W
Total opinions (software + usability)	40	37	37	48	-3	+11
Opinions related to usability	16	12	11	37	-5	+25
Percentage	40%	32%	30%	77%	-10%	+45%

During 1DC the students provided 40 strengths and 37 weaknesses. 16 strengths were related to usability issues (40%). In addition, they provided 12 weaknesses (32%). In the 2DC the students provided 37 strengths and 48 weaknesses. In this case, 11 strengths were related to usability issues (30%) and 37 weaknesses were related to usability (30%). After the conduction of the usability evaluation the strengths related to usability decreased 10% whilst the weaknesses increased 45%.

The results of the relative importance given by the participants to software or usability matters (form F2) confirmed their perception about strengths and weaknesses. After the usability evaluation, the students' opinions changed in order to consider the usability as more important. It seems that usability becomes more relevant for students after they conduct the usability evaluation. These results are presented in Table 2.

Table 2. Strengths (S) and weaknesses (W) related to usability before and after conducting usability evaluation.

Facts	F2			
	1 DC		2 DC	
	Related to software	Related to usability	Related to software	Related to usability
Favorable opinions	37	8	30	15
Percentage	82%	18%	67%	33%

3.2 Detailed understanding of usability.

In Table 3, we present the strengths and weaknesses provided by students in 1DC and 2DC, which are related to usability. This table also includes the variation presented in these aspects after the usability evaluation.

Table 3. Strengths (S) and weaknesses (W) related to usability before and after conducting usability evaluation.

Student	F1				Variance	
	1 DC		2 DC			
	S	W	S	W	S	W
A1	1	2	1	6	0	+4
A2	1	1	0	6	-1	+5
A3	1	2	2	6	1	+4
A4	3	4	0	4	-3	0
B1	2	1	1	4	-1	+3
B2	4	0	3	3	-1	+3
B3	2	1	2	3	0	+2
B4	1	1	1	2	0	+1
B5	1	0	1	3	0	+3
Total	16	12	11	37	-5	+25

The change in the students' opinions between the 1DC and the 2DC, can be grouped into three categories: reduction in the number of strengths and an increase in the number of weaknesses (we identified this category as 'expected change'), no change in the number of strengths and weaknesses (identified as 'no change') and increase in the number of strengths (we identified this category as 'unexpected change').

In the first case, an increase of weaknesses and a reduction of strengths related to usability, present a clear pattern in the change of opinion. After the evaluation, the students changed their opinions in order to report more weaknesses and a lower level of strengths related to usability in their software. The most representative change was given in the high number of weaknesses related to usability reported after the evaluation. For instance, in 1DC the student A-2 provided only one weakness related to usability: "looks awful", although in 2DC, the same student provided six new ones, e.g. "some counterintuitive stuff", "not enough buttons in specific windows", "same labels names – different actions", "confusing interface", "not enough label information" and "not enough indication of selected stuff". Other students also changed their opinions in an important way. This was the case for student A-3 who provided 2 weaknesses in 1DC, but after the usability evaluation, gave 6 weaknesses, e.g. "not consistent in all menus", "dropdown menu blocks buttons", "search function hard to find", "button names can be misleading", "some buttons are missing" and "windows too small". In 2DC, the same student also repeated this last weakness ("windows too small"). A lower variation in weaknesses was presented when it came to the change of opinion of student B-3. First, during 1DC, this student gave only one weakness: "slow UI between normal & full screen". Following this, in 2DC, the student provided three new weaknesses, e.g. "the learning curve", "full screen design flawed" and "bad keyboard navigation".

There were some cases where the students did not change the number of strength and weaknesses related to usability in their software. For example, student B-3 provided two strengths during 1DC such as "non-distracting design" and "intuitive design". In 2DC this student seemed to maintain his emphasis on the design matter; at that moment he reported two strengths, e.g. "smooth playback" and "nice design in normal mode (not full screen)".

Finally, there was an unexpected change in strengths. Student A-3 provided an additional strength after 2DC. In 1DC this student provided only one strength related to usability: "detailed overview for each entry". In 2DC the student maintained the same strength and gave another: "easy to learn". This student has broken the pattern related to reducing the strengths and increasing the weaknesses associated with usability.

3.3 Detailed results on the relative importance of usability

Our study also collected data relating to the relative importance which the students gave to software and usability matters, before and after their conduction of usability evaluation. These data were collected using the form designed to measure the relative importance given by the students to software/usability concepts (form coded as F2).

These results allow us to see the change in the understanding of usability from another perspective. Our interest was to identify whether or not the students placed more importance on usability matters after conducting the usability evaluation, and if there was a change, how this change occurred.

In this part of our study, we identified two main changes. The first change occurred when the students changed their opinion in order to prefer more usability matters. This change was coded as 'X->U'. On the other hand, the second change occurred when the students had selected software matters such as more important. This alternative change was coded as 'X->S'. Finally, our study also identified one case where no change occurred. We triangulated these results with the students' opinions related to their strengths and weaknesses of their software in order to verify consistency in the results. In Table 4 we present details of these changes.

Table 4. Detailed changes in the relative importance given by the students to software/usability matters, after conducting usability evaluation. (P# Pair of concepts)

Student	P1	P2	P3	P4	P5
A1					
A2	X->U		X->U		
A3			X->U		X->S
A4					X->S
B1	X->U				
B2	X->U				
B3			X->U		
B4	X->U			X->U	
B5		X->S		X->U	X->U

After conducting usability evaluation the students changed their opinion with the aim of considering usability matters as more important. These changes were particularly evident in Group B (students of computer science). Conversely, the group with more change of opinions towards technical aspects of the software, was Group A (students of software engineering). Finally, the common changes of opinion made to place more importance on the usability matter, were oriented to aspects related to designing GUIs and how to apply paradigms which could help this design.

3.4 Patterns in the understanding of usability

After identifying and understanding the improvement in the students' understanding of usability, we focused on exploring whether or not it would be possible to identify the detailed characteristics of this improvement process. In order to systematize the identification of the patterns presented in this process, we proceeded to classify the opinions given by students in both 1DC and 2DC. We focused on those opinions which were related to usability, ignoring the opinions coded as technical aspects related to software. Here, both the strengths and the weaknesses are treated together as a unified group of opinions; our interest was to identify the characteristics of the opin-

ions in general, regardless of their nature. The approach of taxonomy of usability proposed in [1], provided us with the framework for the classification. This taxonomy defined six attributes presented in the concept of usability: Knowability (K), Operability (O), Efficiency (E), Robustness (R), Safety (S) and Subjective satisfaction (SS).

In the case of Group A, the opinions are related to the attributes which are more oriented to users (K, O and SS). It is remarkable that the emphasis from students is placed on aspects connected to the "knowability" attribute, especially after the usability evaluation. The "knowability" attribute is defined as "the property by means of which the user can understand, learn, and remember how to use the system" [1]. For example, two weaknesses reported by the students were "Some counterintuitive stuff" and "Not enough indication of selected stuff". In the same way, one of the strengths was "Easy to learn".

This apparent concern of students for the user needs seems to be produced after the usability evaluation rather than at the same time. During the interviews that we held with two members of this group, their opinions seemed not to show a special affinity by the user during the evaluation. When we asked the students what they were thinking when they saw the users during the tests, one student said "... it can be quite funny to see users operate your program, especially when you make some easy task like finding a button, something that they may find difficult because your program may have some design issues". Another student, reflecting on a specific mistake that all the users found, reported that he "felt embarrassed because in the case of the mistake, it was an obvious mistake, never mind that the users found others mistakes too." More specifically, when we inquired about some special feeling of students toward the users during the tests, the first student responded: "Not really, just found it a bit hilarious, because our design was flawed". The second student reported: "I don't remember to have any specific feeling for the users; I just tried to be as objective as I could. I just focused taking notes all the time".

Next, we showed the students the information provided by them during 1DC and 2DC. We also showed them the change presented in their opinions between those DCs. At this time we asked them if they had realized, at the time of the 2DC, that their change of opinion was more oriented to usability. The first student stated: "Not sure if I was aware of it or not. Might have been since we've put a decent amount of effort in correcting our design mistakes afterwards". The other student reported that "Yes, I thought that I was more usability oriented, when I filled this form because I had my eyes open for the usability part of our software. I really notice which things the users felt using our software". Finally, we wanted to know if the students thought that their feelings toward users had been changed after observing the usability evaluations; their answers were categorical. The first student stated "Well yes, I did not take the user into account before, well of course a little bit but not as much. Lesson learned overall, that the user knows how the users want the design, the designer does not". The second one said "Well, I felt thanked for the users for point out the mistakes we made in our software". These partial results confirm that the students recognize the importance of users, that they express a genuine interest in those usability issues more connected to users' needs, and finally, that these feelings seem to be generated after the evaluation.

On the other hand, in the case of Group B again here it is possible to see a clear orientation to “knowability” attributes, e.g. the weakness “Relevant help information on every form” and the strength “Buttons have size compared to how often they are used”. In addition, these students also chose opinions related with the attribute “operability”, defined at the taxonomy as "the capacity of the system to provide users with the necessary functionalities and to permit users with different needs to adapt and use the system”. For example, one of the weaknesses was “The learning curve” whilst one of the strengths was “Easy to use when have been used once”. Finally, the students also selected opinions connected to the attribute “Subjective satisfaction” (e.i. “the capacity of the system to produce feelings of pleasure and interest in users”). In this case, one of the weaknesses and one of the strengths reported by students was, respectively, “Could have had a prettier GUI” and “It looks nice”.

Contrary to the previous group, the students of the Group B distributed their opinions in those attributes more oriented to users (K, O and SS). It could be possible to explain this difference based on the conditions in which students of Group B made their usability evaluation. These students worked with more users who developed more tasks, something that allowed these students find more usability problems.

We also held an interview with one student of this group in order to try to identify when this affinity by users’ needs occurred. The results were quite similar to those obtained in the previous interviews.

In general, all the students’ opinions show two characteristics. First, their opinions are oriented toward usability attributes and fully oriented to users’ needs. Second, after conducting usability evaluations, this phenomenon increases, specifically with regards to the concern of the students for aspects related to the needs of the users when it comes to understanding, learning, and remembering how to use the software. In Figure 1 we present these results.

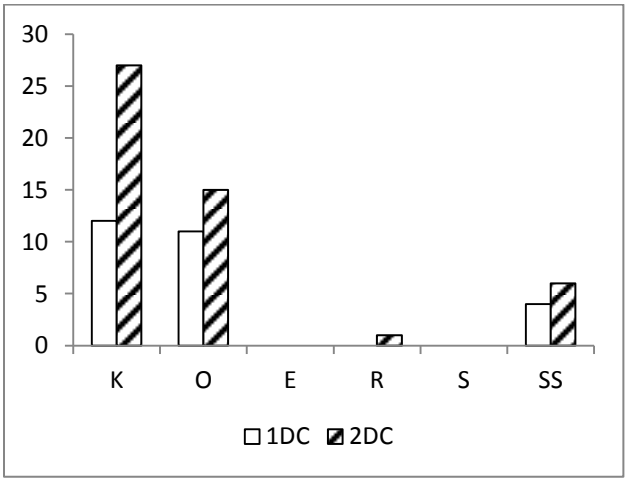

Fig. 1. Distribution of all students’ opinions regarding the usability of their software, before and after conducting usability evaluation.

4 Discussion

The lack of understanding regarding usability is a factor which limits the application of usability activities as, for example the application of usability evaluations in software development [2, 3]. In our study, the lack of understanding regarding usability is represented in those initial perceptions of students about usability in their software. Before the usability evaluation, this perception was characterized by a lower number of weaknesses related to usability (32%). However, after the usability evaluation, developers changed dramatically this opinion and have reported numerous weaknesses (77%). There is also a change in strengths after the usability evaluation. This initial measurement of the status of the students' understanding of usability, is another example of the low level of relevance that developers normally give to usability matters due, among other things, to their different aims, motivations, or mindset [3], [11], [22]. However, after conducting the usability evaluation, the new measurement of the status of the understanding allowed us to identify a new different perspective held by the students. The corrective action used (usability evaluation) allowed students to gain a different perspective of their software: the users' perspective. At that moment, they could identify new problems in their software (i.e. usability problems or even other functional problems). More important is the fact that their perspective changed in order to realize the relevance of other usability matters. Additional evidence of this change in students' perspective is presented in the analysis of the importance which students gave to usability matters. After conducting the usability evaluation the students changed their opinion, placing more importance on usability matters. These changes were particularly evident in Group B (students of computer science).

This increase in the students' understanding of usability is connected to their empathy toward users, which was increased during the usability evaluation [10]. This is something that we also found in our study when we saw students focusing more on usability issues, after the usability evaluation. This general predilection for usability more than for other technical issues, allows us to infer more attention on users' needs. In addition, analysing the pattern in the understanding of usability allows us to identify that the students certainly had, but more important yet, have increased their attention to usability matters which are strictly connected with users' needs (i.e. knowability, operability, and subjective satisfaction).

Some students (Group A) emphasized their opinions in the knowability attribute. Others spread their opinions on all the attributes connected with users' needs. This could be explained in the characteristics of each usability evaluation. Students of Group B interacted with more users who made more tasks; more usability problems were found during this process. These students worked more time with the users consequently, this higher level of interaction with them allowed students to have a wider vision of users' needs.

The reinforcement in the pattern of the understanding about usability, generated after conducting usability evaluation, suggest some affectation of the students as a result of the observation of the users interacting with their software. This does not occur simultaneously at the same time as the interaction with the users.

The interviews with some of the students clearly allowed us to identify that during the moment of the evaluation, they were not focused on the users. Their concerns at that moment were more of a personal nature. This is the case for one student, who was in charge of conducting the evaluation and expressed his concern because the users had problems thinking out loud. Other student found it funny that users could not use the software system well due to some design flaws, or finally the case of the user who felt embarrassed. All these feelings are strictly personal. Furthermore, all students were conclusive in affirming that during the process they had no special feeling toward the users. However, evidence of empathy is clear when we see the improvement in the students' understanding of usability and the pattern of this understanding. In actual fact, analysing the feelings of students toward users, at the moment of the interviews, we see only positive thoughts towards them.

This unconscious acquisition of empathy by students is crucial in order to gauge whether the process behind the generation of empathy of the students is the contagion of users' emotions that they experimented with in their interaction with the users. Indeed, this unconscious process is the cornerstone of basic conceptualization of the EC theory [8], [17]. In actual fact, our study confirmed that the students acquired the users' feelings or emotions before generating an emotional empathy and, later the cognitive empathy which is reflected in their opinions during the 2DC and the interviews. These opinions are an example of the eventual affective response identified by [5].

This is not trivial, nor is it an elaborate explanation of a process which may seem very logical. Identifying EC as the source of empathy of students, allows us to realize that there are corrective actions which are more effective than other traditional options (e.g. regular training), in order to improve the understanding of usability. This is the case with the observation or conduction of usability evaluations by software developers. In our experiment we detected a level of understanding about usability at 1DC obtained by students, mainly as a result of the training received, including topics related to HCI. After the usability evaluation, the understanding of usability changed radically. This new level of understanding, and empathy toward the users, was generated by EC as a result of the interaction with the users in more real conditions.

5 Conclusion

In this paper we presented the results of a quasi-experiment conducted in order to explore the origin of novice software developers' empathy toward users and its relation to the improvement process in understanding usability. We explored the status of the understanding of usability before and after a corrective action (conduction of a usability evaluation) made in order to enhance the understanding. The corrective action allowed the participants in our study to interact with users while they were working with a software system. In our study we explored in detail the improvement in understanding usability, in order to identify clues to help us trace the origin of the empathy toward users, produced as a result of this improvement process.

We found a clear enhancement in the understanding of usability after applying the corrective action; we detected a new student perspective when it came to their software and also about the relative importance that they gave to usability matters over other software technical aspects. This change in the students' perspective reflects an impact on what Sohaib & Khan, as well as Lee, have identified as the aims and motivations of developers which are normally present in their mindset. A better understanding of usability should involve a higher level of empathy toward users; something which we explored by studying the patterns presented in the understanding.

Patterns presented in the understanding regarding usability before and after the corrective action draw a picture and thus make it possible to find a clear and generalized preference for those usability attributes fully connected with users' needs, i.e. knowability, operability and subjective satisfaction.

More relevant for us was the confirmation that this empathy towards users was acquired in an unconscious process of contagion generated during the interaction with users; something which is consonant with EC theory.

Our study attempts to fill the gap in the literature by explaining the origin of novice software developers' empathy toward users. Additionally, our research suggests that in any corrective action to improve the understanding of usability, there is something behind the scenes. EC plays a relevant role in these processes. EC theory explains why those actions which involve more interaction with real users, in real conditions, could have better results that other more traditional actions, such as training.

Considering that our results could only be generalized to novice software developers, it is necessary to conduct more longitudinal studies in order to explore how EC interacts with other kinds of software developers.

Acknowledgments

The research behind this paper was partly financed by National University (Costa Rica), Ministery of Science and Technology – MICIT (Costa Rica), National Council for Scientific and Technological Research - CONICIT (Costa Rica), and the Danish Research Councils (grant number 09-065143).

References

1. Alonso-Ríos, D., Vázquez-García, A., Mosqueira-Rey, E., & Moret-Bonillo, V. (2009). Usability: a critical analysis and a taxonomy. International Journal of Human-Computer Interaction, 26(1), 53-74.
2. Ardito, C., Buono, P., Caivano, D., Costabile, M. F., Lanzilotti, R., Bruun, A., & Stage, J. (2011). Usability Evaluation: A Survey of Software Development Organizations. In Proceedings of 33 International Conference on Software Engineering & Knowledge Engineering. Miami, FL, USA.
3. Bak, J. O., Nguyen, K., Risgaard, P., & Stage, J. (2008, October). Obstacles to usability evaluation in practice: A survey of software development organizations. In Proceedings of the 5th Nordic conference on Human-computer interaction: building bridges (pp. 23-32). ACM.

4. De Vignemont, F. (2004). The co-consciousness hypothesis. Phenomenology and the Cognitive Sciences, 3(1), 97-114.
5. Decety, J., & Jackson, P. L. (2006). A social-neuroscience perspective on empathy. Current directions in psychological science, 15(2), 54-58
6. Gilmore, D. J., & Velázquez, V. L. (2000, April). Design in harmony with human life. In CHI'00 Extended Abstracts on Human Factors in Computing Systems (pp. 235-236). ACM.
7. Grudin, J. (1991). Obstacles to user involvement in software product development, with implications for CSCW. International Journal of Man-Machine Studies, 34(3), 435-452)
8. Hatfield, E., Cacioppo, J. T., & Rapson, R. L. (1994). Emotional contagion. Cambridge Univ Pr.
9. Hertel, G., Niedner, S., & Herrmann, S. Motivation of software developers in Open Source projects: an Internet-based survey of contributors to the Linux kernel. Research policy, 32(7), (2003), 1159-1177.
10. Hoegh, R. T., Nielsen, C. M., Overgaard, M., Pedersen, M. B., & Stage, J. (2006). The impact of usability reports and user test observations on developers' understanding of usability data: An exploratory study. International journal of human-computer interaction, 21(2), 173-196.
11. Lee, J. C. (2006). Embracing agile development of usable software systems. In CHI'06 extended abstracts on Human factors in computing systems (pp. 1767-1770). ACM.
12. Newell, A. F., Morgan, M. E., Gregor, P., & Carmichael, A. (2006, April). Theatre as an intermediary between users and CHI designers. In CHI'06 Extended Abstracts on Human Factors in Computing Systems (pp. 111-116). ACM
13. Patton, J. (2002, November). Hitting the target: adding interaction design to agile software development. In OOPSLA 2002 Practitioners Reports (pp. 1-ff). ACM
14. Rasch, R. H., and Tosi, H. L. Factors affecting software developers' performance: an integrated approach. MIS quarterly, (1992), 395-413.
15. Rosenbaum, S., Rohn, J. A., & Humburg, J. (2000, April). A toolkit for strategic usability: results from workshops, panels, and surveys. In Proceedings of the SIGCHI conference on Human factors in computing systems (pp. 337-344). ACM.
16. Rubin, J., & Chisnell, D. (2008). Handbook of usability testing: how to plan, design and conduct effective tests. John Wiley & Sons.
17. Schoenewolf, G. (1990). Emotional contagion: Behavioral induction in individuals and groups. Modern Psychoanalysis, 15(1), 49-61
18. Seffah, A., & Metzker, E. (2004). The obstacles and myths of usability and software engineering. Communications of the ACM, 47(12), 71-76.
19. Shadish, W. R., Clark, M. H., & Steiner, P. M. (2008). Can nonrandomized experiments yield accurate answers? A randomized experiment comparing random and nonrandom assignments. Journal of the American Statistical Association, 103(484), 1334-1344.
20. Singer, T., & Lamm, C. (2009). The social neuroscience of empathy. Annals of the New York Academy of Sciences, 1156(1), 81-96
21. Skov, M. B., & Stage, J. (2012). Training software developers and designers to conduct usability evaluations. Behaviour & Information Technology, 31(4), 425-435.
22. Sohaib, O., & Khan, K. (2010). Integrating usability engineering and agile software development: A literature review. In Computer Design and Applications (ICCDA), 2010 International Conference on (Vol. 2, pp. V2-32). IEEE.
23. William R.. Shadish, Cook, T. D., & Campbell, D. T. (2002). Experimental and quasi-experimental designs for generalized causal inference. Wadsworth Cengage learning.

Integrating Usability Evaluations into Scrum: A Case Study Based on Remote Synchronous User Testing

Fulvio Lizano[1], Maria Marta Sandoval[2], Jan Stage[1]

[1]Aalborg University, Department of Computer Science, Aalborg, Denmark
{fulvio,jans}@cs.aau.dk
[2]National University, Informatics School, Heredia, Costa Rica
msandova@una.cr

Abstract. The tendency to empower users in the software development process encourages the continuing search for ways to reconcile the interests of agile methodologies and Human-Computer Interaction (HCI) activities. The practice of agile methods, e.g. Scrum, is normally focused on high productivity, sometimes leaving aside other important aspects of software development such as usability. On the other hand, HCI methods usually attempt to reach solid conclusions through extensive and formal studies, which can consume significant resources and time. In this paper we present an instrumental single case study which offers an example of how usability evaluations can be integrated into a Scrum project by using Remote Synchronous User Testing (RS). Our approach suggests that the RS process should be conducted by the same developers who integrate the developing team. Our results indicate that RS can be used as a strategy to efficiently and easily integrate usability evaluations into Scrum projects. The most valuable benefit obtained in this integration is related to the opportune feedback offered by usability testing, which can be incorporated to the developing process immediately as is provided through agile principles. Other elements of our approach could help solve other problems normally present in other efforts made in order to integrate usability evaluations into agile methods. The major problem in our case study was related to the difficulty presented by software developers in terms of changing their usual focus when they have to conduct usability evaluations.

Keywords: Software development, usability evaluation, Remote Synchronous User Testing (RS), SCRUM, integrating RS into SCRUM.

1 Introduction

Scrum, as well another Agile methodologies, emphasizes simplicity and speed in the software development process [4]. This explains why these methodologies have become popular in numerous organizations [12].

However, simplicity and speed could make any integration of these development methods with formal usability techniques (e.g. usability evaluations) difficult. Considering the valuable feedback obtained in usability evaluations [9], there is an in-

adfa, p. 1, 2011.

creasing interest in the integration of these kinds of tests into Scrum and other agile methods.

In this paper we present an instrumental single case study [15], [24] which offers an example of how to integrate usability evaluations into a Scrum project.

The case study has two aims. Firstly, considering the Scrum iterative approach, we are interested in exploring which usability evaluation activities/artifacts should be used throughout the process. Secondly, considering that we propose extensive participation from software developers, we want to explore the implications that such participation has in the integration, mainly in terms of how the developers' focus changes during the integration.

In the second section of this paper we present the related works. Next, we present the method used in this case study. After that, we describe the methodological approach for integrating usability evaluations into Scrum. The paper subsequently describes and analyzes the main results of our case study. The remaining sections cover discussion and conclusions, which includes future works proposals.

2 Related works

Notwithstanding the interest in integrating usability into agile methods, literature is prolific in terms of references to obstacles to achieving this purpose [2, 3]. In the particular case of Scrum, there are deeper differences originating from the foundations of both approaches. Usability is focused on how users will work with the software, whereas agile development is centered on how software should be developed [23].

It is possible to identify different viewpoints regarding the integration between usability evaluation and Scrum. Sohaib and Khan [23] identified several tensions between usability and agile methods: the agile approach is characterized by a focus on the client more than the user, it develops functional software that is not necessarily useful, there is an emphasis on acceptation and unit testing more than usability testing, and finally User-Centered Design (UCD) is not normally a priority in agile projects. Lee and McCrickard [18] identified the origin of these tensions in the differing aims and motivations of Software Engineering (SE) and HCI practitioners, combined with a significant quantity and variety of techniques and methodologies existing in both fields.

It is a fact that the differences of aims between SE and HCI practitioners negatively affect the aforementioned integration. There are different perspectives about what is important in software development [17]. SE practitioners focus on designing, implementing and maintaining software, minimizing the relevance of human-computer interfaces. On the contrary, HCI practitioners focus on developing software with high orientation to users in order to allow them to work with the soft-ware effectively.

This dissimilarity of goals could lead to a lack of collaboration between software developers and HCI practitioners. Jerome and Kazman [11] found that SE and HCI practitioners do not closely collaborate with other professionals outside their knowledge areas. The limited collaboration tends to occur too late in the software development process, which reduces its effectiveness.

Finally, the lack of formal application of HCI and SE methods is another factor that could complicate integration. For example, Jia [12] found that only the 38% of participants in her study reported using Scrum "by the book". This issue could explain why the UCD community complains about the limited application of some agile principles (e.g. individuals and interactions over processes and tools, and customer collaboration over contract negotiations) [21].

Several solutions have been suggested in order to integrate usability in software development process. Sohaib and Khan [23] proposed increasing the iterative approach and testing throughout the lifecycle, adopting usability activities by integrating user scenarios and including usability practitioners in agile teams.

Alternatively, Fischer [8] proposes an integration approach based on international standards, which are the result of the consensus of experts, by providing consistency, repeatability of processes, independence of organizations, quality and facilities for communication.

Coincidentally, Ferré, Juristo and Moreno [7] proposed the integration of HCI activities into software projects by using SE terminology for HCI activities in order to allow developers to understand HCI concepts. The Ripple framework could be used as an example of such an approach [19]. This framework proposes an event-driven design representation that offers developers and HCI practitioners a common framework to represent artifacts generated at each stage of SE and HCI lifecycles.

Other practical considerations were recommended by Hussain et al. [10], e.g. smaller tests with iterations to test only certain parts of the software and using smaller groups of 1-2 users into each usability evaluation.

3 Method

The case study was developed considering a small core system designed to manage the data resulting from the supervision process of undergraduate system engineering students' projects. The supervision process is carried out by regular professors of the system engineering courses.

We decided to define a case protocol based on the widely accepted Yin theory [24] for case studies. Our case protocol included specifications such as period of time for the case study, location, hypothesis, research questions, unit of analysis, data analysis plan and also a proposed outline for the future paper.

Data collection is based on several theoretical approaches. Our case protocol primarily considers the documents resulting from the tests (e.g. usability plan, user tasks, usability final report and guidelines) [13], [20]. We also used some Scrum documents [14] and other secondary sources of data collection (e.g. emails, personal interviews, a focus group meeting and videos of sessions).

The analysis used the general analytic strategy of relying on theoretical propositions [24]. We mainly focused on the data resulting from usability evaluations in order to contrast the results of the case study against other findings reported in the literature.

Data evaluation was focused on assessing usability reports delivered during the integration process. We used a checklist based on Capra approach [6]. Additionally, the data collected was triangulated with results obtained in personal interviews and a focus group meeting [16].

4 Integrating usability evaluations into SCRUM: A proposal for a methodological approach

Our proposal of integration is based on usability evaluations created by Remote Synchronous User Testing (RS) [1]. We organized the tests in an iterative scheme of smaller usability evaluations [10]. In each sprint, we made a small usability evaluation by using only two users. Each user participated in a single test and we recruited different users in each test. We also used a few tasks related to specific parts of the software (normally 4-5 tasks per test). The software developers were in charge of conducting the usability evaluation [5], [22].

This approach allowed a simple and practical integration. By using RS in an iterative scheme, we achieved enough test coverage. More importantly, conduction of the evaluations by developers allowed them to easily realize and understand the main usability problems present in their software.

5 Case description and analysis

This section covers the description and analysis of the case study. The first subsection covers the first actions that took place at the beginning of the case. The remaining two sub-sections present the project and the main usability evaluation issues. Although the aim of the study is to explore how integrate usability evaluations into a Scrum project, the description of the case study will focus on usability evaluations.

5.1 Prologue: first actions in the case.

It is possible to consider a simple spreadsheet with crude data related to students' projects as the foundations of the project. The professors had used this spreadsheet as a master record of projects, students, contacts, etc. The evident limitations related to security and the lack of control over this data management provides adequate reasons for developing a software system that could make the management process of the data related to the students' projects more secure and practical. Because of the urgency of the project, there was a consensus to develop the software using an agile methodology, specifically Scrum. In addition, among other factors, the usability of the final software product was cited as a highly desirable feature.

In early August 2012, a web-conference was made between the authors of this study, the professors' coordinator and members of the Scrum team. In this web-conference, we took several decisions. Firstly, bearing in mind resource limitations regarding staff, the project's roles were defined considering the product owner (the

professors' coordinator), three developers and one usability adviser (co-author of this paper). Secondly, we decided on the software tools that should be used in the project. The final accord was related to the definition of the project's schedule.

5.2 The project.

An initial definition of the project considered three sprints with several main activities. The sprints were coded as '0', '1' and '2'. Sprint '0' was focused on definition-planning matters. After finished sprint '2', together with the product owner we decided to add an additional sprint coded as '3'. Our intention was explore an improvement of some usability artifacts.

5.3 Usability evaluations.

Sprint '0' was used in definition-planning issues. The main activities were related to defining the team and roles during the different sprints, defining the preliminary soft-ware architecture facts, defining the business value project, and finally other activities more connected to the usability evaluations that will be conducted by RS (i.e. the settings of RS software tools and training).

Starting from sprint '1', the developers conducted several usability evaluations. The tests included sessions with two users and a final analysis session conducted by a facilitator. In Figure 1 a session with users (section A) and the final analysis session (section B) can be observed.

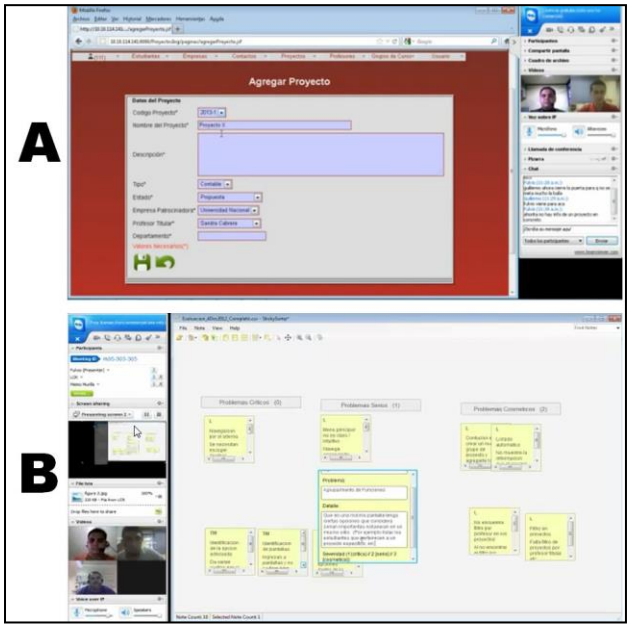

Fig. 1. A: User test session. B: the final analysis session

In Table 1 is possible to see the main results of the case study. In the table it is also possible see the main artifacts designed for the tests, which were based on Rubin and Chisnell [20]. In sprint '0' a usability plan was defined along with several usability tasks and guidelines. Only in the case of the usability plan and the guidelines for the remote analysis final session was a second release of such documents necessary, specifically during sprint '2'. In sprint '0', no usability evaluation was conducted. Some of the usability tasks also required a second release. The main data from the usability evaluations and the assessment of the usability problem reports is also presented in the table.

Table 1. The case study main facts.

Activity	Deliverable / results	Sprint 0	Sprint 1	Sprint 2	Sprint 3
Usability evaluation artifacts	Usability Plan	1-R1		1-R2	
	User Tasks	T1-R1 T2-R1 T3-R1 T4-R1 T5-R1 T6-R1 T7-R1	T1-R1 T2-R1 T3-R1 T4-R1	T1-R1 T4-R1 T5-R1 T6-R1 T7-R1	T1-R2 T4-R2 T8-R1 T9-R1
	Usability Problems Report	0	1	1	1
	Test-Monitor guideline	1			
	Logger guideline	1			
	Remote Analysis Final Session guideline	1-R1		1-R2	
Test facts	Average per user/task		220,75	300,7	210,25
	Critical usability problems			3	2
	Serious usability problems		1	3	2
	Cosmetic usability problems		6	1	1
Assessment of Usability Problem Report	Final result		6,5	9	9

6 Discussion

This case study presents an example of how to integrate usability evaluations into a Scrum project. One of the aims of the study was related to exploring which usability

evaluation activities/artifacts should be used during the process. The other aim is connected to the effects of developers' participation in such integration. Considering that we proposed extensive participation from developers, we were interested in exploring the implications that such participation has in the integration, mainly in terms of how the developers' focus changes during the integration. In the following sub-sections, we will discuss these research questions in more detail.

6.1 Usability evaluation activities/artifacts considered in the integration

The results of our study confirmed that Scrum emphasizes simplicity and speed [4] and can be compatible with usability evaluations by using RS. The iterative schema of Scrum allows for the implementation of several usability evaluations on a software system, something that was also proposed by Sohaib and Khan [23]. Practical usability evaluations can be conducted by using smaller tests on certain elements of the software with smaller groups of users as was suggested by Hussain et al. [10]. In the case study we conducted short evaluations in one day where smaller groups of only two users performed a small number of tasks (i.e. 4-5). Several test iterations with different users carrying out different tasks allowed us to achieve adequate testing coverage.

This iterative approach allows a set of usability activities/artifacts that could change from one sprint to another to be used. Thus, it is possible to use only the strictly necessary activities or artifacts in every sprint, e.g. designing a single usability plan which could be upgradeable if necessary, defining several usability tasks for the entire project, etc.

The fact that practical tests can be conducted in Scrum by using the strictly necessary forms, guidelines, etc. makes integration feasible, practical and easy. The usability feedback which is obtained as a direct result of the usability evaluations fits perfectly into the agile principles [21]. Indeed, the limited application of Scrum [12] could be improved through the aforementioned integration.

6.2 Changing the focus in the Usability Evaluation iterations.

Our study found that the main challenge of the integration between usability evaluation and Scrum was to change the focus of software developers in the usability evaluations conducted in the iterations. During the first evaluation, the developers who participated in usability evaluation had forgotten or simply ignored the specific rules documented in the guidelines. This attitude, although apparently unintentional, could be another manifestation of the software developers' mindset [2, 3]. During the second evaluation, the re-training efforts produced good results; audio video recording, notes and the results of closing meetings confirmed that the problem had decreased.

The problem can be handled by using guidelines and giving developers training that emphasizes not only the techniques required to conduct the tests but also those specific rules related to how they should conduct themselves during such tests. These guidelines, which are based on Rubin and Chisnell approach [20], implement the Fischer approach [8] regarding international standards in the integration. The use of

guidelines and the iterative schema of usability evaluations were easily understood by developers as these elements are well-known to them. This is another example of the integration proposed by Ferré, Juristo and Moreno [7] through using SE terminology for HCI activities in order to facilitate developers' understanding of HCI concepts.

Changing the focus of developers is a complex issue due to the developers' mindset [2, 3] and the difference of perspectives between developers and HCI practitioners [18], [23]. However, even considering the limitations of this case study, it is possible to identify certain improvements in developers' focus regarding usability matters when they are interacting with usability activities, especially in an iterative scheme of usability evaluations. This improvement process will definitely reduce the lack of collaboration between SE and HCI practitioners [11].

7 Conclusion

Usability evaluations conducted using RS can be integrated into Scrum projects, re-gardless of the formal assumptions presented in this evaluation method. RS does not create major obstacles for the Scrum principles of simplicity and speed. In this single instrumental case study, we proposed a methodological approach that considers using RS conducted by software developers in Scrum projects as a way to integrate usability evaluations into this agile method. Throughout the case we identified the specific RS activities or instruments involved in the Scrum's sprints. We also studied changes in software developers' focus when they conducted usability evaluations and the major implications of the interchanging roles system used by developers in the Scrum's sprints.

Our approach to integrating RS into Scrum presents several interesting advantages. Firstly, the iterative strategy required by the Scrum dynamic makes it unnecessary to use all the activities and artifacts normally used in a usability evaluation process. Secondly, it is relatively easy for developers to conduct usability evaluations by using RS; the use of guidelines and basic training has confirmed the feasibility of such an aim. Finally, because this integration has shown to be practical, it is possible to use a similar approach in order to increase the chances of applying usability evaluations in other agile methods. This case study has shown how it is possible to handle some problems presented in the efforts to integrate usability evaluations into agile methods, e.g. the inherent tensions found between these approaches, the difference in aims and the lack of collaboration between the SE and the HCI practitioners, the lack of a for-mal Scrum application and usability evaluation approaches, etc.

The main problems present in this integration are related to the changes of focus of developers when they have to conduct usability evaluation. Despite the fact that we implemented some correcting actions relatively successfully, it seems that this issue could affect the quality of the evaluation results, at least in the first ones. Fortunately, the approach used in this case of study and the iterative dynamic which is presented in the Scrum method allow the problem to be detected in a relatively short period of time in order to adjust the guidelines, training, etc.

Future works include developing a more extensive longitudinal study in order to explore, over longer periods of time, the effectiveness and sustainability of the corrective actions used in this case study. This study should use a more extensive number of cases that could include different sizes and compositions of software development teams.

Acknowledgments

The research behind this paper was partly financed by National University (Costa Rica), Ministery of Science and Technology – MICIT (Costa Rica), National Council for Scientific and Technological Research - CONICIT (Costa Rica), and the Danish Research Councils (grant number 09-065143).

References

1. Andreasen, M. S., Nielsen, H. V., Schrøder, S. O., & Stage, J. (2007, April). What happened to remote usability testing?: an empirical study of three methods. In Proceedings of the SIGCHI conference on Human factors in computing systems (pp. 1405-1414). ACM
2. Ardito, C., Buono, P., Caivano, D., Costabile, M.F., Lanzilotti, R., Bruun, A., Stage, J.: Usability Evaluation: a survey of software development organizations. In Proceedings of 33 International Conference on Software Engineering & Knowledge Engineering. Miami, FL, USA (2011).
3. Bak, J.O., Nguten, K., Risgaard, P., Stage, J.: Obstacles to Usability Evaluation in Practice: A Survey of Software Development Organizations. In Proceedings of the 5th Nordic conference on Human-computer interaction: building bridges, pp.23-32. ACM, New York, NY, USA (2008).
4. Beck, K., Beedle, M., Van Bennekum, A., Cockburn, A., Cunnimgham, W., Fowler, M., ... & Thomas, D. (2001). Principles behind the agile manifesto. Retrieved, 11, 2008.
5. Bruun, A., and Stage, J. Training software development practitioners in usability testing: an assessment acceptance and prioritization. In *Proc. OzCHI*, ACM Press, (2012). 52-60.
6. Capra, M. G. (2006). Usability problem description and the evaluator effect in usability testing (Doctoral dissertation, Virginia Polytechnic Institute and State University).
7. Ferré, X., Juristo, N., Moreno, A.: Which, When and How Usability Techniques and Activities Should be Integrated. In: Seffah, A., Gulliksen, J., Desmarais, M.C. (eds.) Human-Centered Software Engineering - Integrating Usability in the Software Development Lifecycle. Human-Computer Interaction Series, vol. 8, Kluwer, Dordrecht (2005)
8. Fischer, H. (2012, June). Integrating usability engineering in the software development lifecycle based on international standards. In Proceedings of the 4th ACM SIGCHI symposium on Engineering interactive computing systems (pp. 321-324). ACM.
9. Hoegh, R. T., Nielsen, C. M., Overgaard, M., Pedersen, M. B., & Stage, J. (2006). The impact of usability reports and user test observations on developers' understanding of usability data: An exploratory study. International journal of human-computer interaction, 21(2), 173-196.
10. Hussain, Z., Lechner, M., Milchrahm, H., Shahzad, S., Slany, W., Umgeher, M., ... & Wolkerstorfer, P. (2012, January). Practical Usability in XP Software Development Pro-

cesses. In ACHI 2012, The Fifth International Conference on Advances in Computer-Human Interactions (pp. 208-217).
11. Jerome, B., & Kazman, R. (2005). Surveying the solitudes: An investigation into the relationships between human computer interaction and software engineering in practice. Human-Centered Software Engineering—Integrating Usability in the Software Development Lifecycle, 59-70
12. Jia, Y. (2012). Examining Usability Activities in Scrum Projects–A Survey Study (Doctoral dissertation, Uppsala University).
13. Kjeldskov, J., Skov, M. B., & Stage, J. (2004, October). Instant data analysis: conducting usability evaluations in a day. In Proceedings of the third Nordic conference on Human-computer interaction (pp. 233-240). ACM
14. Kniberg, H. (2007). Scrum and XP from the Trenches. InfoQ Enterprise Software Development Series
15. Lazar, J., Feng, J. H., & Hochheiser, H. (2010). Research methods in human-computer interaction. Wiley.
16. Lazar, J., Feng, J. H., & Hochheiser, H. (2010). Research methods in human-computer interaction. Wiley
17. Lee, J. C. (2006, April). Embracing agile development of usable software systems. In CHI'06 extended abstracts on Human factors in computing systems (pp. 1767-1770). ACM.
18. Lee, J. C., & McCrickard, D. S. (2007, August). Towards extreme (ly) usable software: Exploring tensions between usability and agile software development. In Agile Conference (AGILE), 2007 (pp. 59-71). IEEE.
19. Pyla, P., Pérez-Quiñones, M., Arthur, J., & Hartson, H. (2005). Ripple: An event driven design representation framework for integrating usability and software engineering life cycles. Human-Centered Software Engineering—Integrating Usability in the Software Development Lifecycle, 245-265.
20. Rubin, J., & Chisnell, D. (2008). Handbook of usability testing: how to plan, design and conduct effective tests. John Wiley & Sons
21. Seffah, A., Desmarais, M.C., and Metzker, E. HCI, Usability and Software Engineering Integration: Present and Future. In Seffah, A. et al. (eds.) Human-Centered Software Engineering. Springer: Berlin, Germany, 2005.
22. Seffah, A., Metzker, E.: The obstacles and myths of usability and software engineering. Commun. ACM, 47, pp 71-76, December (2004).
23. Skov, M. B., & Stage, J. (2012). Training software developers and designers to conduct usa-bility evaluations. Behaviour & Information Technology, 31(4), 425-435.
24. Sohaib, O., & Khan, K. (2010, June). Integrating usability engineering and agile software development: A literature review. In Computer Design and Applications (ICCDA), 2010 International Conference on (Vol. 2, pp. V2-32). IEEE
25. Yin, R.K. (2003). Case Study Research: Design and Methods (Thousand Oaks, CA: Sage).

Increasing Empathy toward Users' Needs by using Usability Evalulations: A field-experiment.

Fulvio Lizano (corresponding author), Jan Stage

Department of Computer Science, Aalborg University, Aalborg, Denmark
fulvio@cs.aau.dk (corresponding author), jans@cs.aau.dk

Abstract. Although HCI techniques as usability evaluations are considered strategic in software development, there are important obstacles in the application thereof. One of the most relevant is the software developers' lack of understanding regarding usability. This problem is connected to the lack of empathy toward users' needs on the part of software developers. Empathy is strategic in order to handle this and other problems presented in both HCI and SE. This paper presents results from an empirical study in which we explored the effectiveness of some typical usability evaluation methods to improve of empathy towards users' needs. The results show that the classical, user-based thinking aloud protocol significantly increases empathy. The alternative remote synchronous method is also effective and is more efficient, considering its logistical and economic advantages. Both methods create a scenario in which users work in conditions that are more realistic and expose their implicit needs while being observed by developers. This interaction allows the contagion of users' emotions by software developers, which increases their empathy toward users' needs.

Keywords: Empathy towards users' needs; understanding regarding usability; emotional contagion; usability evaluation.

1 Introduction

There is clarity regarding the relevance of a high level of usability in software. However, many software developers reveal contradictory behavior in that do not give priority to aspects of usability. This lack of priority is rooted in the different aims and motivations of SE and HCI practitioners (Lee 2006, Sohaib and Khan 2010). It is also possible to link this issue to what some researchers have called the software developers' mindset (Bak et al. 2008, Ardito et al. 2011). In the particular case of usability evaluations, the developers' lack of priority is related to a lack of understanding when it comes to the concept of usability concept (Rosenbaum 2000). This problem has become a serious

obstacle to applying usability in the software development process (Bak et al. 2008, Ardito et al. 2011, Seffah, A., and Metzker 2004).
Recent studies have suggested that involving software developers in the conduction or observation of usability evaluations with users could increase their understanding of usability (Bruun, A., and Stage 2012, Hoegh et al. 2006, Skov and Stage 2012). In parallel with this, it is possible to identify an improvement in empathy towards users' needs (Hoegh et al. 2006).
A clear understanding of reasons why empathy increases in a usability evaluation context is not trivial. This paper presents the results of an empirical study that aimed to explore the effectiveness of several typical usability evaluation methods to improve or increase the software developers' empathy towards users' needs. In the following sections, we offer an overview of related works, the method used in our research, the results of our study, the analysis and conclussions.

2 Related work

Empathy is the capacity of an observer to observe, identify, and understand another person's feelings, producing in the observer, at least partially, a sense of sharing the same feelings as the other person (Decety and Jackson 2006, Singer and Lamm 2009). Empathy occurs in the context of an emotional contagion processes (De Vignemont 2004, Singer and Lamm 2009).
Emotional Contagion (EC) theory has been extensively used in group psychology. According to EC theory, some of the key points presented in the contagion process are the tendency to automatically mimic and synchronize expressions, vocalizations, postures and movement, converging emotionally (Hatfield et al. 1994) and creating an unconscious induction of emotional states and behavioral attitudes (Schoenewolf 1990). This process involves the perception of emotions using nonverbal signals, such as facial expressions, body language and tone rather than words (Barsade 2002).
Several behavioral processes are connected to EC theory. Normally, these processes are inherent in social influences and could occur both consciously and subconsciously (Barsade 2002). The observer assumes a submissive role in her/his interaction with the observed who, in turn, assumes a dominant role. The particular circumstances or personalities of each are decisive in establishing who assumes a particular role (Schoenewolf 1990). In this process, emotions may be passed to others in order to influence not only their emotions but also their perceptions of other aspects. For example, Pugh (2001) argues that when employees display positive empathy towards customers, the customers perceive more quality in the service obtained.
Empathy and EC theory have been considered in HCI research (Mattelmäki and Battarbee 2002; Yammiyavar 2005; Preece 2001). Empathy towards users on the part of software developers has also been part of the research agenda in HCI and SE. Gilmore and Velázquez (2000) found that developers' participation in user experiences is important in order to improve the software developer's empathy for users. Newell et al. (2006) also reported the relevance of the generation of empathy with users during the process of interface design. Furthermore, empathy is relevant in software development in general (Grudin 1991; Patton 2002; Karn et al. 2007).
Despite the aforementioned studies, there is a gap in the literature when it comes to training-based strategies to improve software developers' empathy towards users' needs. Some HCI training proposals (LeBlanc et al. 2006, Lunt et al. 2008) do not consider empathy toward users' needs. A marginal reference regarding some efforts to increase the empathy of software engineering students was provided by Shaw and Dermoudy (2005) who questioned the effectiveness of traditional approaches based on lectures to increase empathy.
An alternative approach, based on software developers participating in or observing usability evaluations, has emerged in some recent studies. Hoegh et al. (2006) found an improvement in the

lack of understanding regarding usability and a parallel increase in empathy towards users' needs by software developers who observed usability evaluations. Gilmore and Velázquez (2000) also argued the relevance of interaction with users in order to increase empathy towards them. The feasibility of conducting usability evaluations by developers was proved by Skov and Stage (2012). Bruun and Stage (2012) argued that, with only a few hours of training, software practitioners could be taught to identify usability problems.

Many of the usability evaluation methods used in these studies correspond to classical methods that are regularly used, such as conventional thinking aloud protocol (Rubin and Chisnell 2008), inspection methods like heuristic evaluation (Nielsen and Molich 1990), and methods that are more modern, like the remote synchronous testing model (Andreasen et al 2007).

Despite these research efforts, no specific studies have been conducted on exploring the effectiveness of these typical usability evaluation methods in order to improve or increase the software developers' empathy towards users' needs.

3 Method

We conducted a field experiment aimed at exploring the effectiveness of several usability evaluation methods in order to improve or increase the software developers' empathy towards users' needs.

We used a between-group design with four conditions corresponding to four types of usability evaluation:

- Classical inspection method based on heuristic methods (hereinafter referred to as Heuristic).
- Variation of the inspection method with "supervision" (hereinafter referred to as Heuristic-Supervised). We introduced this variation in order to provide additional support to evaluators and to compensate for support received by participants in other conditions.
- Classical laboratory-based think-aloud method (hereinafter referred to as Lab).
- Modern remote synchronous testing method (hereinafter referred to as Remote).

3.1 Participants

A total of 36 people participated in our study. Participants were advanced System Engineering students who were organized in 12 teams of 2-4 members. The average age was 22.2 (SD =2.17) and 11% were female. In addition to the courses taken previously, the participants had amassed nearly 18 months of real experience of practical academic activity by developing a software system in a real organization, with real users and their needs. The scope of all these software projects were carefully controlled in order to guarantee similar characteristics in quality/size/complexity.

Of the nearly 30 existing projects, 16were pre-selected as potential participants in the experiment. These pre-selected projects belonged to organizations which could have at least 3 potential users available and were willing to participate in the study. After that, we randomly selected the number of organizations and participants needed for the experiment. Finally, again by using a random distribution, we grouped the actors into the different conditions used in the experiment. In each condition, each team evaluated another team's software. The software project were developed in

diverse organizations (e.g., schools, colleges, biological research organizations, municipal police stations, etc.). As an incentive for participation, the participants received extra credits.

3.2 Training

All participants received training and advice from the researchers of this study during the experiments (the Remote condition's participants were contacted in a remote way). The training included a workshop by developing experiences such as planning, conducting and reporting usability evaluations. Participants received specific instructions in order to consider three categories of the usability problems identified: critical, serious, and cosmetic (Andreasen et al. 2007). The training was based on the forms and guidelines defined for the study (Nielsen and Molich 1990, Kjeldskov et al 2004, Rubin and Chisnell 2008). The number of hours spent in training was 10.

3.3 Procedure

We conducted two data collections (1DC and 2DC). 1DC was conducted approximately one month before the study. Following this, the participants worked in teams to design and conduct the usability evaluations according to the corresponding conditions. We separated each test in two main parts. The first part, under the responsibility of the team who made the software, corresponded to the planning of the complete process. The second part was the conduction of the test itself under the responsibility of another team. In this way, we guaranteed that all the teams participated impartially enough in a comprehensive testing process. In Table 1 we present the settings and specific procedure followed in each condition. The second data collection, 2DC, was conducted immediately after the finalization of the tests. In addition, we held four focus group sessions (1 per condition) which were audio-visually recorded in order to transcribe and analyze them later.

Condition	*Settings*	*Procedure*
Heuristic	• Classical heuristic evaluation (Nielsen and Molich 1990)	• Preliminary individual assessment. • Final Result group
Heuristic-Supervised	• Similar to Heuristic condition.	• Similar to Heuristic condition. • Heuristics' interpretation support by "supervisor"
Lab	• State-of the-art usability lab. • Conventional thinking aloud protocol (Rubin and Chisnell 2008)	• 3 Users sessions/5 Tasks each • Test-Monitor sat next to user. Logger and observers took notes. • Audio-Visual recording. • Final analysis session (Kjeldskov 2004)
Remote	• Remote synchronous testing (Andreasen 2007) • All participants separated spatially.	• Similar to Lab condition.

Table 1: Settings and procedure used in the conditions

Usability evaluations was only to set an environment where the participant could interact or not, with users in a more realistic context. In this paper We do not include unnecessary details about how the usability evaluations where conducted.

3.4 Data collection and analysis

Data collection in this study was focused on the participants' opinion about their software. Our intention was to identify if such opinions were connected or not, to usability matters in order to explore the participants' level of understanding of usability. Any change in this understanding implies an improvement in empathy towards users' needs (Gilmore and Velázquez 2000, Hoegh et al. 2006).
We used two forms in each data collection. The first form had the aim of allowing the participants to express their opinions related to the main strong and weak points of the software. This form was coded as 1DC-F1 and 2DC-F1, respectively. The second form had the goal of measuring the relative importance given by the participants to certain software/usability concepts. We measured the relative importance by using several pairs of sentences that could illustrate several activities/concepts related to HCI or SE.
We conducted our analysis of the first form following a three-step procedure. First, the concepts contained in the forms were reviewed and clarified. Second, we coded the concepts related to usability matters. Finally, the remaining concepts were coded as technical aspects related to software. With this, we analyze variations in the participants' understanding regarding usability between 1DC and 2DC. In the second form we calculate the level of relative importance reported by each participant. We analized this results individually in order to triangulate with the previous concepts given on the other forms. Thus, we have identified and measured the improvement of the participants' understanding regarding usability. We also used independent-sample t tests and paired-sample t tests. Finally, the data collected during the focus group sessions were quantified by using basic principles of the grounded theory approach by Strauss and Corbin (Strauss and Corbin 1998).

4 Results

4.1 Initial empathy status: initial overall understanding of usability

When the study began, we were interested in gauging the perceptions of participants of all conditions regarding their understanding of usability. This understanding is directly related to the use of concepts connected to usability or software. In Table 2, we present the general results obtained. First, we present the results obtained when we enquired about the strong (ST) and the weak (W) points of the software (F1). We organized these opinions into two categories: strong points and weak points. In each category, we present the results obtained for the points related to usability (U) and for the points related to software (S). In both cases, these are expressed in terms of quantity and percentage (#/%). In addition, we present the relative importance that participants placed on usability/software matters (F2) in terms of percentages (%).

Condition	F1				F2	
	Strong points (#/%)		*Weak points (#/%)*		*(%)*	*(%)*
	U	*S*	*U*	*S*	*U*	*S*
Heuristic	13 38%	21 62%	3 14%	19 86%	34%	66%
Heuristic-Supervised	6 19%	26 81%	3 13%	20 87%	29%	71%
Lab	5 15%	29 85%	7 30%	19 73%	27%	73%
Remote	13 35%	24 65%	7 29%	18 71%	47%	53%

Table 2: Results at 1DC. F1: Strong / weak points related to usability (U) / software (S) matters. F2: relative importance

The results regarding relative importance confirmed the perception of the strong and the weak points of the software. Initially, the participants' opinions were mainly focused on technical software matters.

In general, usability matters were considered only marginally by participants. This situation is interesting, considering that, at the time of the study, the participants had had contact with users for almost 18 months and that they had pursued numerous computing courses for seven semesters. This last factor explains their preference for software matters.

Considering our interest in usability, the independent-sample t test suggests that there is no significant difference between strong (ST) and weak (W) points for almost all conditions, except in the cases of the strong points of Heuristic versus Lab and for the strong points of Lab versus Remote (see Table 3 for the results of these tests).

	Heuristic	*Heuristic-Supervised*	*Lab*	*Remote*
Heuristic		(ST) p=.056	(ST) (p=.048	(ST) p=1.000
Heuristic-Supervised	(W) p=1.000		(ST) p=.806	(ST) p=.056
Lab	(W) p=.129	(W) p=.129		(ST) (p=0.48)
Remote	(W) p=.129	(W) p=.129	(W) p=1.00	

Table 3: Independent-sample t test, df=16 for strong (ST) / weak (W) points related to usability, at 1DC. (p) =significant

In general, it is possible to affirm that the participants of all conditions had a similar overall understanding of usability before the usability evaluations. The results reveal that, before the evaluations, the understanding of usability was relatively the same for all participants in our study, which is logical considering that they have the same background (training, courses pursed, contact with users). Similarly, considering the relationship between understanding and empathy (see the related work section), these results suggest that at the beginning of the study the participants had a similar level of empathy towards users' needs.

4.2 Change in the empathy towards users' needs: Improvement of understanding of usability

After the usability evaluations, we identified changes in the understanding of usability. The intensity of these changes was variable in the different conditions. Table 4 presents the general results obtained in 2DC in the four conditions and the variations with respect to 1DC. In this table, we present only the results related to usability matters. First, we present the strong and weak points (F1) and the results obtained at 2DC for the strong points expressed in terms of quantity and percentage (#/%). Next, we present the variation of this aspect with respect to 1DC (expressed in terms of a percentage - %). Thereafter, we present the results obtained for the weak points, which are expressed in terms of quantity and percentage (#/%). Finally, we present the variation of the weak points with respect to 1DC (expressed in terms of a percentage - %). In addition, we present the relative importance (F2). First is the percentage obtained at 2DC and the variation with respect to 1DC (in both cases, this is expressed in terms of percentages - %).

Condition	*F1*				*F2*	
	Strong. (#/%)	*Var. (%)*	*Weak (#/%)*	*Var. (%)*	*(%)*	*Var. (%)*
Heuristic	10 35%	-3%	3 16%	+2%	44%	+10 %
Heuristic-Supervised	11 38%	+19 %	8 30%	+17 %	42%	+13 %
Lab	25 71%	+56 %	12 50%	+23 %	60%	+33 %
Remote	27 63%	+28 %	9 50%	+21 %	68%	+21 %

Table 4: Results at 2DC. F1: Strong / weak points related to usability (results and variation respect 1DC). F2: relative importance of usability (results and variation respect 1DC).

These results allow the identification of an improvement in the understanding of usability after the usability evaluations. This improvement is more evident in Lab and Remote conditions. After evaluations, the participants of such conditions had a different perception of the relative importance of usability in respect of software matters (33% and 21% of increment respectively for Lab and R). These results were confirmed by the perception regarding the strong and weak points related to

usability, at 2DC. The participants of Lab condition reported 35 strong points, of which 71% were related to usability (56% more than 1DC). The same participants also increased their perception of weak points related to usability (50%, 23% more than 1DC). For Remote condition, it is possible to observe the same situation: 63% strong points (28% more than 1DC) and 50% weak points (23% more than 1DC).

In order to confirm if the improvement in the understanding of usability was only presented in Lab and Remote conditions, paired-sample t tests were made in order to identify significant differences between 1DC and 2DC, for strong and weak points in the four conditions. Significant differences were detected for strong points at Lab and Remote conditions (see Table 5).

Condition	*Strong points*	*Weak points*
Heuristic	p=.282	p=1.000
Heuristic-Supervised	p=.139	p=.051
Lab	(p=.002)	p=.247
Remote	(p=.011)	p=.512

Table 5. Paired-sample t test, df=8 for strong and weak points (1DC vs. 2DC). (p) =significant.

In addition, the independent-sample t test suggests that, at 2DC, there is a significant difference between the strong points for Lab-Remote conditions when compared to Heuristic with Heuristic-Supervised conditions. The same situation occurred with the weak points of Lab condition versus Heuristic condition. Finally, no significant differences were detected between the strong or the weak points for Lab versus Remote conditions, and neither between the strong or the weak points for Heuristic versus Heuristic-Supervised conditions (see Table 6). This absence of significant differences confirms that Lab and Remote conditions changed at the same time. Final results in these conditions (at 2DC) confirm a change of a similar magnitude. The same parallelism is present in Heuristic and Heuristic-Supervised conditions, with the difference that no significant difference at 2DC means that no changes occurred in either condition.

	Heuristic	*Heuristic-Supervised*	*Lab*	*Remote*
Heuristic		(ST) p=.807	(ST) (p=.002)	(ST) (p=.003)
Heuristic-Supervised	(W) p=.091		(ST) (p=.008)	(ST) (p=.009)
Lab	(W) (p=.026)	(W) p=.343		(ST) p=.714
Remote	(W) p=.063	(W) p=.779	(W) p=.490	

Table 6: Independent-sample t test, df=16 for strong (ST) / weak (W) points related to usability, at 2DC. (p) =significant

In summary, the change in Lab and Remote conditions allowed us to confirm that, after the usability evaluations, only the participants of such conditions have significantly increased their understanding of usability and, consequently, their empathy towards users' needs.

4.3 Nature of the improvement of understanding of usability: Confirming the increase of empathy

More references to usability matters in Lab and Remote conditions imply that participants focused more on usability after their participation in the evaluations and, consequently, there was an improvement in the understanding of usability for such conditions. At the focus group sessions held in these conditions, we identified increased concern/attention on usability; specifically, on users' roles and their needs.

Some examples of the opinions provided by participants during the focus group could better explain this situation. Most of them highlighted the importance of users in the development process, "The system should be usable enough for users. If not, what would be its purpose?" or

"I must confess that in the past, I have developed software only thinking in that final result should be a functional enough software. I had never thinking in simplicity or attractive for users".

In the focus group session for the Heuristic-Supervised condition, we also detected some concern and interest on users' issues. However, in the case of Heuristic condition, usability matters were almost entirely overlooked. Just before finishing the focus group session for Heuristic condition, we requested a final reflection on the software development process. The participants only provided concepts related to the importance of the overall process, the courses, the experience obtained, and the prospects for their future careers.

5 Discussion

5.1 Empowering empathy for users: The usability evaluation approach

The initial understanding regarding usability identified in 1DC confirms the differences of the aims and motivations of SE and HCI practitioners (Lee 2006, Sohaib and Khan 2010), the software developers' mindset (Bak et al. 2008), and the resistance to User-Centered Design or Usability (Rosenbaum et al 2000). Before the usability evaluations, the participants showed more focus on software matters. This level of understanding also reflected low levels of empathy regarding users' needs.

The level of empathy of participants was modified after the usability evaluations. We detected more identification on the part of the participants with usability concerns, something that is fully consistent with previous studies (Gilmore and Velázquez 2000, Hoegh et al. 2006). The change was more evident and significant in those evaluations conducted under Lab and Remote conditions. Because there were no significant differences at 2DC between strong points related to usability in such conditions (see Table 6), it is possible to argue that the resulting level of understanding regarding usability and empathy were similar for such conditions. We confirmed this during the focus group sessions.

The viability of conducting usability evaluations by software developers (Bruun and Stage 2012, Skov and Stage 2012) was also confirmed in our study. The predilection of participants for the strong points of their software can be explained by the developers' sense of individual ability and a personal

identification with the software (Rasch, Tosi 1992; Hertel et al. 2003). In addition, this is another manifestation of the software developers' mind-set (Bak et al. 2008, Ardito et al. 2011).
Classical usability evaluation methods based on thinking aloud protocol (usability lab and remote synchronous testing) are effective for increasing/improving empathy toward users' needs. However, considering the statistically similar results obtained in both methods, it is a fact that the remote synchronous testing method should be considered to be the best option for the software development process; logistical considerations also make this option sufficiently attractive.
Tentative increments in Heuristic and Heuristic-Supervised conditions are not significant but are always interesting. Even considering that no interaction with users occurred in these methods, results suggest that the participants' understanding and empathy were affected as a result of the assessment of other software.
We cannot contrast our results with existing literature. No previous studies have focused on proposals aimed at improving or increasing software developers' empathy towards users' needs. HCI training proposals (Rasch and Tosi 1992, Mattelmäki and Battarbee 2002) do not consider this issue. Alternatives used in other fields could be considered (Boker et al. 2004). However, similar to other fields (Stepien and Baernstein 2006), the diversity of definitions of empathy and the inadequacy of empathy measurement instruments could produce problems when applying this approach.
Indeed, in the context in which our study was conducted, we found some evidence that suggested the limited effectiveness of traditional approaches based on training. The initial level of the participants' empathy was based on their previous training (courses that included some HCI topics) and their practical experience in technical matters, interaction with users, and so on. However, ultimately, usability evaluations were more effective by producing a relevant improvement in empathy. In the usability evaluation context, it is possible to potentiate reflection to and focus on users' feelings. Furthermore, improving empathy towards users is another way to improve soft skills in software developers, as such soft skills which are normally less well supported in university pedagogy (Begel and Simon 2008, Taft 2007).

5.2 Processes involved in improving/increasing empathy toward users' needs

The statistically similar results in the increase of understanding and empathy presented in Lab and Remote conditions imply the existence of similar mechanisms behind these results, particularly those of the EC theory.
EC theory involves the automatic or unconscious contagion of users' emotions (Schoenewolf 1990, Hatfield et al. 1994). In our study, the unconscious contagion experienced by participants who saw the users working with a software system in the usability evaluation context explains the change of perspective of the participants. This was confirmed during the focus group sessions.
The absence of interaction with users in Heuristic and Heuristic-Supervised conditions explains why, in these conditions, participants did not experience similar contagion of users' emotions.
Another EC theory concept presented in our experiment is related to the definition of who will be the observer and who the observed in the contagion process. The particularity of the circumstances or personalities involved is a key factor in this regard (Schoenewolf 1990). In our experiment, participants observed users working with a software system. Not only did the participants assume the role of observers, but the users also assumed an active role by thinking aloud, expressing their thoughts, opinions and feelings. The developers passively received these emotions. Specific guidelines and protocols prevented other interactions that could have modified this scenario.

Finally, EC theory has also established that the remote contagion of emotions is possible (Hancock et al. 2008, Kramer 2012). Our study confirmed an increase in empathy not only in a usability lab, but also remotely and with significant resource savings.

Identifying EC theory's mechanisms in the context of our study is not trivial. EC theory helps to understand the reasons behind the high efficiency and effectiveness of conduction or observation of usability evaluations by software developers in order to increase their empathy toward users' needs. This is an approach that should be seriously considered seriously.

6 Conclusion

This paper provides an empirical exploration focused on the potential of usability evaluations to increase or improve empathy toward users' needs. The main contribution of this study is its empirical basis in a realistic software development context.

The usability evaluation methods that included interaction with users had remarkable results in the generation of empathy towards users' needs. Both methods used in our study (usability lab and remote synchronous testing) provided vivid interaction with users, a scenario that enabled the contagion of users' emotions by software developers. Practical and logistical considerations make the remote synchronous testing method the best option to be considered in the software development process.

Our study has a main limitation. The participants in the study were final year undergraduate students. Nevertheless, real conditions present in our study have allowed control of this bias.

Future work may be oriented towards conducting further longitudinal studies in order to explore results in various contexts and with developers who are more experienced.

Acknowledgments

The research behind this paper was partly financed by UNA-MICIT-CONICIT (Costa Rica) and the Danish Research Councils (grant number 09-065143). We are very grateful to the participants, observers, facilitators, organizations and users that helped us in this research

References

Andreasen, M. S., Nielsen, H. V., Schrøder, S. O., and Stage, J., 2007. What happened to remote usability testing?: an empirical study of three methods. *In Proc. SIGCHI*, ACM Press, 1405-1414.

Ardito, C., Buono, P., Caivano, D., Costabile, M.F., Lanzilotti, R., Bruun, A., and Stage, J., 2011. Usability Evaluation: a survey of software development organizations. *In Proc. SEKE*, 282-287.

Bak, J.O., Nguten, K., Risgaard, P., and Stage, J., 2008. Obstacles to Usability Evaluation in Practice: A Survey of Software Development Organizations. In *Proc. NordiCHI*, ACM Press, 23-32.

Barsade, S. G., 2002. The ripple effect: Emotional contagion and its influence on group behavior. *Administrative Science Quarterly*, 47(4), 644-675

Begel, A., and Simon, B., 2008. Novice software developers, all over again. In *Proc. ICER*, ACM Press, 3-14.

Boker, J. R., Shapiro, J., and Morrison, E. H., 2004. Teaching empathy to first year medical students: evaluation of an elective literature and medicine course. *Education for Health*, 17(1), 73-84.

Brunero, S., Lamont, S., and Coates, M., 2010. A review of empathy education in nursing. *Nursing Inquiry*, 17(1), 65-74

Bruun, A., and Stage, J., 2012. Training software development practitioners in usability testing: an assessment acceptance and prioritization. *In Proc. OzCHI*, ACM Press, 52-60.

De Vignemont, F., 2004. The co-consciousness hypothesis. *Phenomenology and the Cognitive Sciences*, 3(1), 97-114.

Decety, J., and Jackson, P. L., 2006. A social-neuroscience perspective on empathy. *Current directions in psychological science*, 15(2), 54-58.

Fagerholm, F., and Münch, J., 2012. Developer Experience. *In Proc. ICSSP*, IEEE, (2012),73-77.

Gilmore, D. J., and Velázquez, V. L., 2000. Design in harmony with human life. Ext. Abstracts CHI 2000, ACM Press, 235-236.

Grudin, J., 1991. Obstacles to user involvement in software product development, with implications for CSCW. *International Journal of Man-Machine Studies*, 34(3), 435-452.

Hancock, J. T., Gee, K., Ciaccio, K., and Lin, J. M. H., 2008. I'm sad you're sad: emotional contagion in CMC. *In Proc. ACM conference on Computer supported cooperative work*, ACM Press, 295-298.

Hatfield, E., Cacioppo, J. T., and Rapson, R. L., 1994. *Emotional contagion*. Cambridge University Press

Hertel, G., Niedner, S., and Herrmann, S., 2003. Motivation of software developers in Open Source projects: an Internet-based survey of contributors to the Linux kernel. *Research policy*, 32(7), 1159-1177.

Hoegh, R. T., Nielsen, C. M., Overgaard, M., Pedersen, M. B., and Stage, J., 2006. The impact of usability reports and user test observations on developers' understanding of usability data: An exploratory study. *International journal of Human-Computer Interaction*, 21(2), 173-196.

Karn, J. S., Syed-Abdullah, S., Cowling, A. J., and Holcombe, M., 2007. A study into the effects of personality type and methodology on cohesion in software engineering teams. *Behaviour & Information Technology*, 26(2), 99-111

Kjeldskov, J., Skov, M. B., and Stage, J., 2004. Instant data analysis: conducting usability evaluations in a day. *In Proc. NordiCHI*, ACM Press, 233-240.

Kramer, A. D., 2012. The spread of emotion via facebook. In Proc. SIGCHI, ACM Press, 767-770.

LeBlanc, R. J., Sobel, A., Diaz-Herrera, J. L., and Hilburn, T. B., 2006. *Software Engineering 2004: Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering*. IEEE Computer Society Press.

Lee, J. C., 2006. Embracing agile development of usable software systems. *Ext. Abstracts CHI 2006*, ACM Press, 1767-1770.

Lunt, B. M., Ekstrom, J. J., Gorka, S., Hislop, G., Kamali, R., Lawson, E., and Reichgelt, H., 2008. *Curriculum guidelines for undergraduate degree programs in information technology*. IEEE Computer Society Press.

Mattelmäki, T., and Battarbee, K., 2002. Empathy probes. In *Proc. PDC*, 266-271.

Newell, A. F., Morgan, M. E., Gregor, P., and Carmichael, A., 2006. Theatre as an intermediary between users and CHI designers. *Ext. Abstracts CHI 2006*, ACM Press, 111-116.

Nielsen, J., and Molich, R., 1990. Heuristic evaluation of user interfaces. *In Proc. SIGCHI*, ACM Press, 249-256.

Patton, J., 2002. Hitting the target: adding interaction design to agile software development. *In OOPSLA 2002 Practitioners Reports*, ACM Press, 1-ff.

Preece, J., 2001. Sociability and usability in online communities: determining and measuring success. *Behaviour & Information Technology*, 20(5), 347-356.

Pugh, S. D., 2001. Service with a smile: Emotional contagion in the service encounter. *Academy of management journal*, 44(5), 1018-1027.

Rasch, R. H., and Tosi, H. L., 1992. Factors affecting software developers' performance: an integrated approach. *MIS quarterly*, 395-413.

Rosenbaum, S., Rohn, J. A., and Humburg, J., 2000. A toolkit for strategic usability: results from workshops, panels, and surveys. *In Proc. SIGCHI*, ACM Press, 337-344.

Rubin, J., and Chisnell, D., 2008. *Handbook of usability testing: how to plan, design and conduct effective tests*. John Wiley & Sons.

Schoenewolf, G., 1990. Emotional contagion: Behavioral induction in individuals and groups. *Modern Psychoanalysis*, 15(1), 49-61.

Seffah, A., and Metzker, E., 2004. The obstacles and myths of usability and software engineering. *Communications of the ACM*, 47(12), 71-76.

Shaw, K., and Dermoudy, J., 2005. Engendering an empathy for software engineering. *In Proc. of the 7th Australasian conference on Computing education*-Volume 42, 135-144.

Singer, T., and Lamm, C., 2009. The social neuroscience of empathy. *Annals of the New York Academy of Sciences*, 1156(1), 81-96.

Skov, M. B., and Stage, J., 2012. Training software developers and designers to conduct usability evaluations. *Behaviour & Information Technology*, 31(4), 425-435.

Sohaib, O., and Khan, K., 2010. Integrating usability engineering and agile software development: A literature review. *In Proc. Computer Design and Applications ICCDA 2010 International Conference on*, Vol. 2, IEEE, 32-38.

Stepien, K. A., and Baernstein, A., 2006. Educating for empathy. *Journal of general internal medicine*, 21(5), 524-530.

Strauss, A., and Corbin, J. 1998. *Basics of qualitative research. Techniques and Procedures for Developing Grounded Theory*. 2nd ed. SAGE Publications.

Taft, D. K., 2007. Programming Grads Meet a Skills Gap in the Real World. *eWeek.com*.

Yammiyavar, P., 2005. Emotion as a Construct in HCI. *In Research Symposium*, 8.

Usability Evaluations for Everybody, Everywhere:

A field study on Remote Synchronous Testing in Realistic Development Contexts

Fulvio Lizano
Department of Computer Science
Aalborg University
Aalborg, Denmark
fulvio@cs.aau.dk

Jan Stage
Department of Computer Science
Aalborg University
Aalborg, Denmark
fulvio@cs.aau.dk

***Abstract*—Although Human-Computer Interaction (HCI) techniques, as usability evaluations, are considered strategic in software development, there are diverse economic and practical constraints in their application. The integration of these tests into software projects must consider practical and cost-effective methods such as, for instance, the remote synchronous testing method. This paper presents results from a field study in which we compared this method with the classic laboratory-based think-aloud method in a realistic software development context. Our interest in this study was to explore the performance of the remote synchronous testing method in a realistic context. The results show that the remote synchronous testing method allows the identification of a similar number of usability problems achieved by conventional methods at a usability lab. Additionally, the time spent using remote synchronous testing is significantly less. Results obtained in this study also allowed us to infer that, by using the remote synchronous testing method, it is possible to handle some practical constraints that limit the integration of usability evaluations into software development projects. In this sense, the relevance of the paper is based on the positively impact that remote synchronous testing could have in the digital accessibility of the software, by allowing extensive use of usability evaluation practices into software development projects.**

Keywords-Usability evaluations; remote synchronous testing method; integration of usability evaluation in software development projects; field study.

I. INTRODUCTION

Usability has a significant impact on software development projects [15]. Common usability activities, as usability evaluations, are relevant and strategic in diverse contexts (e.g., organizations, software development process, software developers and users) [3], [13].

However, economic and practical issues limit integration of usability evaluations into software projects, where limited schedules and high expectations of stakeholders to obtain effective/efficient results faster, are common. Productivity has been a recurrent concern in the industry [5], [12] and is something that makes it very difficult to justify some HCI activities [20].

Bearing this in mind, any effort to integrate usability evaluations into software projects must necessarily consider practical and cost-effective methods, such as the remote synchronous test.

In this paper, we present the results of a field study that aimed to compare the remote synchronous test method against the classic laboratory-based think-aloud method in a realistic software development context.

In the following section, we offer an overview of related works. The next section presents the method used in our research. Following this, we present the results of our study. After the results are summarized, the paper presents the analysis before concluding with suggestions for future work.

II. RELATED WORKS

Integration efforts of usability evaluations into software projects have economic and practical constraints.

High consumption of resources in usability evaluations is a recurrent perception in diverse contexts [2], [3], [19], [22], [23]. This fact could explain why usability has a lower valuation for the organization's top management [8], becoming manifest by the lack of respect and support for usability and the HCI practitioners [9]. Therefore, cost-justification of usability may be difficult for many companies when it is perceived as an extra cost or feature [20].

On the other hand, three of the most cited practical constraints are related to: the difference of perspectives between HCI and Software Engineering (SE) practitioners, the absence or diversity of methods and, finally, the users' participation.

The first constraint related to the difference of perspectives between HCI and SE practitioners is contextualized in the difference of opinions they have about what is important in software development [17]. This diversity of perspectives results in contradictory points of view regarding how usability testing should be conducted and is something that may result in a certain lack of collaboration between HCI and SE practitioners. It is possible to find the origin of this discrepancy between these two perspectives in the foundations of the HCI and SE fields. Usability is focused on how the user will work with the software, whereas the development of that software is centered on how the software should be developed in a practical an economical way [27]. These conflicting perspectives result in tensions between software developers and HCI practitioners [18], [27].

The second constraint relates to the absence or diversity of methods, and has two opposing views. Firstly, some researchers report a lack of appropriate methods for usability evaluation [2], [19] or a lack of formal application of HCI and SE methods [15]. This situation may explain why the UCD community has expressed criticism about the real application of some software development principles [25]. Secondly, it is reported that the existence of numerous and varied techniques and methodologies in the HCI and SE fields could hamper the integration [18].

Finally, the participation of customers and users has become another relevant limitation for the integration of usability evaluations into software projects [2], [3], [19]. This matter is a permanent challenge to the dynamic of the software development process. Users and customers have their own problems and time limitations, and these normally limit their participation in software development activities such as usability evaluations.

The literature reported different proposals for handling the aforementioned three practical constraints. Firstly, in the case of the difference of perspectives between HCI and SE practitioners, some studies have suggested that increased participation of developers in usability testing could positively impact their valuation of usability [13]. This improvement in the developers' perspectives could make them more conscious of the relevance of HCI techniques.

Secondly, with respect to the absence or diversity of methods, an integration approach based on international standards is proposed [7] in order to enable consistency, repeatability of process, independence of organizations, quality, etc. A similar approach suggests the integration of HCI activities into software projects by using SE terminology for HCI activities [6].

Finally, regarding the constraint related to the participation of customers and users, some researchers have suggested several practical actions (e.g., smaller tests in iterative software development processes, testing only some parts of the software, and using smaller groups of 1–2 users in each usability evaluation [14].

These aforementioned studies were conducted on limited realistic contexts, e.g., literature reviews [7], [20], [23], [25], [27], surveys [2], [5,], [9], [15], [19], experiments in labs [22], [26] and case studies [13], [18]. Other papers cited above present proposals of projects or methods [6], [8], [17]. There are only three studies with a more empirical base in more realistic contexts [4], [13], [14]. Confidence in the results of these studies should be improved by other studies made in a realistic development context.

III. METHOD

We have conducted an empirical study aimed at comparing the remote synchronous testing method (condition R) with the classic laboratory-based think-aloud method (condition L).

By using remote synchronous testing, the test is conducted in real time, but the evaluators are separated spatially from the users [1]. The interaction between the evaluators and the users is similar to those at a usability lab. There are many studies that confirm the feasibility of remote usability testing methods [1], [10], [28]. Actually, there is a clear consensus regarding the benefits obtained by using this method (e.g., no geographical constraints, cost efficiency, access to a more diverse pool of users and similar results as a conventional usability test in a lab) [1], [24]. The main disadvantages are related to problems of generating enough trust between the test monitor and users, a longer setup time, and difficulties in re-establishing the test environment if there is a problem with the hardware or software [1].

Three usability evaluations were made by three teams using a classic usability lab. In addition, another three usability evaluations were conducted by another three teams using a remote synchronous testing method.

All of these teams were formed by final-year students of SE who had 18 months of practical experience working in software development. This experience is the result of an academic project created by the students by developing a software system in a real organization.

A. Participants

In order to be considered for our research, the software projects must meet our requirements regarding users being available for the tests. Considering these criteria, 16 of 30 teams, and their software projects, were pre-selected as potential participants in the experiment. Finally, we randomly selected six teams who were randomly distributed throughout the R and L conditions.

The teams were formed by final-year students who were finishing their last course in System Engineering. These participants were organized into six teams consisting of three members each. A total of 18 people participated in our study. The average age was 22 (SD=2.13) and 17% were female. In addition to the courses taken previously, the participants had amassed nearly 18 months of real experience of practical academic activity by developing a software system in a real organization that sponsored the project. These organizations provided regular assessments and formal acceptance (or rejection) of the software. Several users and stakeholders were also involved in the process. The scope of the software projects was carefully controlled in order to guarantee a similar level of effort from all of the participants. The average of the final assessment of the project was 9.67 on a scale of 1–10 (SD=0.33). As an incentive for participation, the participants received extra credits. The conditions, code, members and software are presented in Table I.

B. Training and advice

All participants received training and advice during the experiments (remotely for R condition). In the training, we presented and explained several forms and guidelines based on commonly used theories [16], [24]. In addition, a workshop was made in order to putting into practice the contents of the training materials. The participants received specific instructions in order to consider three categories of usability problems: critical, serious, and cosmetic [1]. The number of hours spent in training was 10 (four hours in lectures and six hours in practice). Furthermore, the advice provided to the participants included practical issues concerning how to plan and conduct usability evaluations.

TABLE I. TEAMS, MEMBERS, AND STAFF FOR THE USABILITY EVALUATION

Cond.	Code	Members	Software
L	L1	3 males	Students' records in a private college
	L2	1 female, 2 males	Internal postal management system in a financial department of a public university
	L3	1 female, 2 males	Laboratory equipment management in a biological research center belonging to a public university
R	R1	1 female, 2 males	Criminal record in a small municipal police station
	R2	3 males	Management of documents related to general procurement contracts in an official national emergency office
	R3	3 males	Students' records in a public school

C. Procedure

The design of the experiment increased confidence in the results and objectivity of the development teams during the evaluation process. Under the two conditions, each team had to test the software system made by another team, who also tested another software system made by a third team.

Each test had two main parts. The first part, under the responsibility of the team who made the software, corresponded to the planning of the complete process (e.g., planning, checklists, forms, coordination with users, general logistics, etc.). The planning included a session script with 10 potential tasks of the software.

In the second part of the tests, another team conducted the sessions with the users. The test monitor of this team had to select, for each user, five tasks from those previously defined. We thought this measure would increase the impartiality of the process; the developers of the software could not interfere in the selection of the task and the users had to work with different tasks in each session. Next, the test monitor guided the users in the development of the task while the logger and the observers took notes. The test ended with a final analysis session conducted by a facilitator [16].

D. Settings

The test conducted under the L condition used a state-of-the-art usability lab and think-aloud protocol [21], [24]. Each test included three sessions where the users were sat in front of the computer and the test monitor was sat next the users. The logger and observers were present in the same room. In the case of the R condition, the tests were based on the remote synchronous testing [1]. All participants were spatially separated. Users were in the sponsors' facilities. Each test included three sessions with users.

E. Data collection and analysis

Each user session was video recorded. The video included the software session recorder (video capture of screen) and a small video image of the user. Under R conditions, the video also recorded the image of the test monitor. We also used a test log to register the main data of each activity (i.e., date, participant, role, activity and time consumed) and the usability problem reports.

The data analysis was conducted by the authors of this paper based on all data collected during the tests. The tests produced six sets of data for analysis, i.e., six usability problem reports, six test logs and six videos.

The consistency of the classification of the usability problems by participants was one of the main concerns in this study. Consequently, our analysis included an assessment of such classification. Our intention was to be sure that this classification was done consistently according to the instructions given to all participants during the training. We assessed the problem categorization by checking the software directly in order to confirm the categorization given by participants to a usability problem. The videos were thoroughly walked through in order to confirm this categorization.

The tests were conducted on different software systems. There is not a joint list of usability problems. This is the reason why, in our analysis, we compared the differences between both conditions by using average and standard deviations calculated separately for each condition.

Using the test logs, we analyzed the time spent in all the tests. We considered individual and group time consumption. We calculated totals, averages and percentages to facilitate the analysis. We included in this process all the activities made by all members of the teams in the preparation of the test (e.g., usability plan, usability tasks, etc.) and the conducting of the test itself. In the analysis, we also considered other participants, such as the users and observers, in order to consider a more realistic context.

Finally, in order to identify significant differences in the data collected, we used independent-sample t tests.

TABLE II. PROBLEMS IDENTIFIED PER TYPE OF PROBLEM. (%)= PERCENTAGE PER CONDITION.

Cond.-> Problems	L	R
Critical	36 (52%)	33 (56%)
Serious	29 (42%)	22 (37%)
Cosmetic	4 (6%)	4 (7%)
Total	69	59

IV. RESULTS

A. Problems identified per type

Table II shows an overview of the usability problems identified under the two conditions. The problems are classified by their type. The largest number of problems was critical. The lowest number of problems identified was in the category of cosmetic problems. The distribution of all types of problems, among the two conditions, was relatively uniform. An independent-sample t test for the number of usability problems identified for the three categories, under both conditions, showed no significant difference ($p=0.404$). The fact that there are no significant differences between the

TABLE III. USERS' TASKS COMPLETION TIME AND TIME PER PROBLEM. UP= TOTAL NUMBER OF USABILITY PROBLEMS IDENTIFED PER CONDITION

Condition-> Test–User	**L (UP 69)**		**R (UP 59)**	
	Tot. Minutes	**Avg. per task (SD)**	**Tot. Minutes**	**Avg. per task (SD)**
T1–U1	10.8	2.2 (1.9)	30.0	6.0 (1.3)
T1–U2	9.7	1.9 (1.0)	18.3	3.7 (1.6)
T1–U3	12.8	2.6 (2.5)	18.7	3.7 (1.6)
T2–U1	6.1	1.2 (0.4)	17.6	3.5 (1.8)
T2–U2	14.3	2.9 (0.8)	13.3	2.7 (1.3)
T2–U3	8.4	1.7 (0.7)	8.9	1.8 (0.7)
T3–U1	7.4	1.5 (1.0)	11.2	2.2 (2.4)
T3–U2	6.9	1.4 (0.9)	9.0	1.8 (1.4)
T3–U3	11.1	2.2 (1.1)	10.5	2.1 (2.1)
Total Avg. por task (SD)	87.6 1.94 (0.5)		137.4 3.10 (1.3)	
Avg. task completion time per problem, in minutes	1.26		2.32	

L and R conditions is a reflection of the similarity of the effectiveness of these methods in terms of the number of problems identified.

B. Task completion time

The task completion time was less in the tests made under the L condition. In these tests, the users spent a total of 87.6 minutes completing the five tasks assigned to each one. The average time per user/task was 1.94 (SD=0.5). The average task completion time per usability problem identified under the L condition was 1.26. In the tests made under the R condition, the task completion time was 137.4, the average time per user/task was 3.10 (SD=1.3), and the average task completion time per problem was 2.32. In Table III, we present these results.

An independent-sample t test for the task completion time of the nine users considered under the two conditions showed a significant difference (p=0.018).

The analysis of the videos recorded during the tests made under the R condition showed delays due to technical problems – mainly in the communication between the actors (i.e., users, test monitor, technician, etc.). In addition, in general, the users in their normal jobs were more distracted. On the contrary, in the case of the tests made at the laboratory, the users were more focused, and the guidance of the test monitors was more effective.

C. Time spent in the tests

The time spent to complete the tests presents an entirely different perspective to that shown in the previous section. Here, the tests conducted under the R condition consumed less time than that conducted under the L condition.

TABLE IV. TIME SPENT IN THE TESTS. UP= TOTAL NUMBER OF USABILITY PROBLEMS IDENTIFIED PER CONDITION

Condition-> Activity	**L (UP 69)**	**R (UP 59)**
Preparation	2500 (102)	1580 (123)
Conducting test	1320 (73)	840 (42)
Analysis	980 (157)	710 (71)
Moving staff/users	1110 (107)	160 (57)
Tot.time spent per test	5910 (220.5)	3290 (102)
Avg. time per problem in minutes	85.7	55.8

In Table IV, we presented an overview of the time spent in the tests conducted under the two conditions. This table includes the average number of minutes spent on test activities. The standard deviation is shown between parentheses. At the end, the table also shows the average of time per problem in minutes.

These results included all the actors involved in the tests (i.e., users, test monitor, logger, observers, etc.). In this sense, it is possible to consider these results more realistic; here, all of the elements/persons required to perform the tests are included. An independent-sample t test, for the average time spent in the tests, for both conditions, showed an extremely significant difference ($p<0.001$).

The time spent on each activity during the tests confirms these extremely significant differences for all of the activities – except in the analysis. In preparation, conducting the tests, and moving staff, the independent-sample t tests for the time spent in the three tests conducted under each condition, showed extremely significant differences ($p<0.001$ for all of the cases). In the case of the analysis, the difference was significant ($P=0.045$).

V. DISCUSSION

Usability evaluations made by using the remote synchronous testing method are a cost-effective alternative to integrating usability evaluations into software projects. The number of usability problems identified by this method is similar to that obtained by conventional tests made in a usability laboratory. Additionally, there is a significant difference between the time spent on the remote synchronous test method and that spent on the tests made in the lab.

We confirmed the feasibility of conducting usability evaluations by software developers using diverse methods, including the remote synchronous testing method [4], [11], [26]. In addition, we also confirmed the similarity to the number of problems identified by the conventional lab method [1]. However, in the case of the time spent, our results differ from those of others [1] who argue that the time spent to conduct tests by using lab and remote synchronous tests was quite similar. In our case, the difference in time consumption for both methods was significantly favorable in the remote synchronous testing method. A detailed analysis of the test logs showed us that, in the tests made under the L condition, the logistic matters consumed much more time than in the tests under the R condition. Considering our aim of confirming previous findings in a realistic development

context, logistic matters must be considered as factual components of any usability test.

The analysis of the procedures followed the conducting of the tests (reported in the usability problem reports) and the test logs showed that, by using the remote synchronous testing method, it is possible to achieve several practical advantages that save time in the tests.

It is possible to contextualize these advantages in the results of the time spent on the tests' activities shown in Table IV. Firstly, in the case of the preparation activities, the virtualization of the complete coordination process saved time and effort. The coordination between teams and other actors was easier and more efficient by using email, chat, video conferences, etc.

Secondly, in the activities of conducting the tests it was also easy and efficient to use all the software tools used during the tests. Even when considering that the task completion time was shown to be better in the tests made under the L condition (see Table III), differences in the overall process were evident due to this task completion time only being related to the time spent by users to complete the tasks. On the contrary, in the conducting activities of the tests, all of the elements and actors required to conduct the whole test are included (i.e., users, test monitor, logger, observers, etc.)

Thirdly, the difference in the analysis was also significant due to the technological tools that facilitated the conducting of the analysis sessions by the facilitator. In a certain way, the videos also showed that the virtualization of the process seems to produce a shared feeling about the relevance of productivity during the virtual sessions.

Finally, the results in the moving activities explain themselves. In the realistic development context used in this study, it is clear that avoiding the movement of the usability evaluation staff is one of the most relevant advantages in terms of time consumption.

In general, all of the advantages of the remote synchronous test cited in the literature were confirmed in the realistic contexts considered in our study [1], [24]. In the case of the disadvantages, we could only identify – in the analysis of the test logs – some problems in the setting of the hardware and software tools used in the process [1].

At this point in the discussion, the economic advantages of the remote synchronous testing method become evident. Furthermore, this method also helps to handle other practical problems of the integration of usability evaluations into software projects.

In our study, we have also confirmed the feasibility of the active participation of software developers in usability evaluations [4], [13], [26]. The participants played several roles in the usability evaluation teams (e.g., test monitor, logger, observer and technician). This confirmation is relevant when considering the context used in our study (i.e., lab and remote synchronous tests under more realistic conditions). The design of our experiment proved to be very useful because all of the teams actively participated in all of the process (i.e., planning and conducting of the test) and with impartiality. It is a fact that these levels of participation of developers in usability evaluations may impact positively upon their perspective regarding usability and the HCI practitioners [17] and will reduce the tensions between SE and HCI practitioners [18], [27].

Furthermore, in the case of the problem related to the lack of formal application of HCI techniques, our experiment found that by using guidelines and basic training, it is possible to prepare developers for conducting usability evaluations. In a certain way, the theory used to inspire the guidelines used in the tests has followed the suggested approach [7] of using standards to help the integration of usability evaluation into software projects. The analysis of the dynamic of the tests registered in the videos did not show any particular significant problems.

In the case of the tests made by using the remote synchronous testing method, the guidelines were fundamental in conducting the remote process. Considering the similarity of the results in the remote synchronous tests and those obtained in the lab, it is clear that the guidelines served their purpose.

Considering these facts, we can conclude that, by using guidelines based on standards, it is possible to improve the perception of the lack of appropriate methods for usability evaluation [2], [19].

Finally, our study also found that the reported problem [2], [3], [19] relating to the participation of customers and users can be handled well by using the remote synchronous testing method. The users do not need to drastically change their activities. Certainly, the task completion time was higher in the remote synchronous testing method but, putting this element in perspective for the whole process, it is always possible to see the strengths of the remote synchronous testing method. Furthermore, other actors did not have to go to the lab.

VI. CONCLUSSION

In this paper, we presented results of a study aimed to compare the remote synchronous test method against the classical laboratory-based think-aloud method in a realistic software development context. Several tests were conducted by final-year students who had 18 months of practical experience. Although the tests were made on software systems for different organizations and purposes, the scope of these software systems was carefully controlled in order to provide similar settings for the study.

The identification of a similar number of usability problems and lower time consumption, make of Remote Synchronous a good alternative for integrating usability evaluations into software projects. By using this method it is possible to involve more software developers into the conduction of usability testing. Such aim only requires basic training, guidelines and essential advice. Basic guidelines and training allows handling the problems related to the methods. Finally, one of the most relevant advantages of this method is to facilitate the participation of users, developers and other potential actors in the tests. By avoiding unnecessary movements of these persons, their participation will be easily justified

Our study has two main limitations. Firstly, the participants in the study were final-year undergraduate

students. Nevertheless, the real conditions present in our study have allowed for a control of this bias. Secondly, we used only two usability evaluation techniques. However, our selection considered an ideal benchmark of high interaction with users (lab) and the alternative option which was the focus of our study. In our study, we were focused on the problems identified and the time consumption metrics in a realistic development context. For future work, it is suggested that, for the same context, a deeper analysis of other metrics, such as the improvement of the perspective of software developers regarding usability – which is another expected result of close participation of developers in usability evaluations – should be conducted.

ACKNOWLEDGMENT

The research behind this paper was partly financed by UNA, MICIT, CONICIT (Costa Rica), and the Danish Research Councils (grant number 09-065143). We are very grateful to the participants, observers, facilitators, organizations and users that helped us in this research.

REFERENCES

[1] M.S. Andreasen, H.V. Nielsen, S.O. Schrøder, and J. Stage, "What happened to remote usability testing?: an empirical study of three methods," Proc. SIGCHI, ACM Press, 2007, pp. 1405-1414.

[2] C. Ardito et al., "Usability Evaluation: a survey of software development organizations," Proc. 33 International Conference on Software Engineering & Knowledge Engineering, 2011. Pp. 282-287.

[3] J.O. Bak, K. Nguten, P. Risgaard, and J. Stage, "Obstacles to Usability Evaluation in Practice: A Survey of Software Development Organizations," Proc. NordiCHI, ACM Press, 2008, pp.23-32.

[4] A. Bruun and J. Stage, "Training software development practitioners in usability testing: an assessment acceptance and prioritization," Proc. OzCHI, ACM Press, 2012,pp.52-60.

[5] P.F. Drucker, "Knowledge-Worker Productivity: The Biggest Challenge," in California management review,41(2), 1999, pp.79-94.

[6] X. Ferré, N. Juristo, and A. Moreno, "Which, When and How Usability Techniques and Activities Should be Integrated," in Human-Centered Software Engineering - Integrating Usability in the Software Development Lifecycle, Springer Netherlands, 2005, pp. 173-200.

[7] H. Fischer, "Integrating usability engineering in the software development lifecycle based on international standards," Proc. SIGCHI symposium on Engineering interactive computing systems, ACM Press, June 2012, pp. 321-324.

[8] T. Granollers, J. Lorés, and F. Perdrix, "Usability engineering process model. Integration with software engineering," Proc. HCI International, 2003, pp 965-969.

[9] J. Gulliksen, I. Boivie, J. Persson, A. Hektor, and L. Herulf, "Making a difference: a survey of the usability profession in Sweden," Proc. NordiCHI, ACM press, 2004, pp. 207-215.

[10] M. Hammontree, P. Weiler, and N. Nayak, "Remote usability testing," in Interactions, 1, 3, 1994, pp. 21-25.

[11] H.R. Hartson, J.C. Castillo, J. Kelso, and W.C. Neale, "Remote evaluation: The network as an extension of the usability laboratory," Proc. CHI, ACM Press, 1996, pp. 228-235.

[12] A. Hernandez-Lopez, R. Colomo-Palacios, and A. Garcia-Crespo, "Productivity in software engineering: A study of its meanings for practitioners: Understanding the concept under their standpoint," Proc. Information Systems and Technologies (CISTI), IEEE Press, June 2012, pp. 1-6.

[13] R.T. Hoegh, C.M. Nielsen, M. Overgaard, M.B. Pedersen, and J. Stage, "The impact of usability reports and user test observations on developers' understanding of usability data: An exploratory study," in International journal of Human-Computer Interaction, 21(2), 2006, pp. 173-196.

[14] Z. Hussain et al., "Practical Usability in XP Software Development Processes," in Proc. ACHI, January 2012, pp. 208-217.

[15] Y. Jia, "Examining Usability Activities in Scrum Projects–A Survey Study," Doctoral dissertation, Uppsala Univ., 2012.

[16] J. Kjeldskov, M.B. Skov, and J. Stage, "Instant data analysis: conducting usability evaluations in a day," Proc. NordiCHI, ACM Press, 2004, pp. 233-240.

[17] J.C. Lee, "Embracing agile development of usable software systems," In Proc.CHI'06 extended abstracts, ACM Press, 2006, pp. 1767-1770.

[18] J.C. Lee and D.S. McCrickard, "Towards extreme (ly) usable software: Exploring tensions between usability and agile software development," in Proc. Agile Conference (AGILE), IEEE Press, August 2007, pp. 59-71.

[19] F. Lizano, M.M. Sandoval, A. Bruun, and J. Stage, "Usability Evaluation in a Digitally Emerging Country: A Survey Study," Proc. INTERACT, Springer Berlin Heidelberg, 2013, pp. 298-305.

[20] G.H. Meiselwitz, B. Wentz, and J. Lazar, Universal Usability: Past, Present, and Future, Now Publishers Inc., 2010.

[21] J. Nielsen, Usability engineering, Morgan Kaufmann Publishers, 1993.

[22] J. Nielsen, "Guerrilla HCI: Using discount usability engineering to penetrate the intimidation barrier," in Cost-justifying usability, 1994, pp. 245-272.

[23] D. Nichols and M. Twidale, "The usability of open source software," in First Monday, 8(1), [online], Available: http://firstmonday.org/ojs/index.php/fm/article/view/1018/939, [retrieved: 01, 2014], 2003

[24] J. Rubin and D. Chisnell, Handbook of usability testing: how to plan, design and conduct effective tests, John Wiley & Sons, 2008.

[25] A. Seffah, M.C. Desmarais, and E. Metzker, "HCI, Usability and Software Engineering Integration: Present and Future," In Human-Centered Software Engineering, Seffah, A. et al. (eds.), Springer: Berlin, Germany, 2005.

[26] M.B. Skov and J. Stage, "Training software developers and designers to conduct usability evaluations," in Behaviour & Information Technology, 31(4), 2012, pp. 425-435.

[27] O. Sohaib and K. Khan, "Integrating usability engineering and agile software development: A literature review," Proc. ICCDA, IEEE Press, 2010, vol. 2, pp. V2-32

[28] K.E. Thompson, E.P. Rozanski, and A.R. Haake, "Here, there, anywhere: Remote usability testing that works," Proc. Conference on Information Technology Education, ACM Press, 2004, pp. 132–137.